Evolutionary computation based on nanocomposite training: application to data classification

Eleonore Gabrielle Blanche Vissol-Gaudin

Hatfield College
Durham University

A Thesis submitted in fulfillment of
requirements for the degree of Doctor of Philosophy

Durham University

Advanced Materials, Electronics and Communications Research Challenge
Department of Engineering
Faculty of Science
Durham University

March 2020

Dedicated to Richard Wainwright (1940-2019)

Evolutionary computation based on nanocomposite training: application to data classification

Eléonore Gabrielle Blanche Vissol-Gaudin
Submitted in fulfillment of requirements for the degree of Doctor of Philosophy

Abstract

Research into novel materials and computation frameworks by-passing the limitations of the current paradigm, has been identified as crucial for the development of the next generation of computing technology. Within this context, evolution in materio (EiM) proposes an approach where evolutionary algorithms (EAs) are used to explore and exploit the properties of un-configured materials until they reach a state where they can perform a computational task. Following an EiM approach, this thesis demonstrates the ability of EAs to evolve dynamic nanocomposites into data classifiers. Material-based computation is treated as an optimisation problem with a hybrid search space consisting of configuration voltages creating an electric field applied to the material, and the infinite space of possible states the material can reach in response to this field. In a first set of investigations, two different algorithms, differential evolution (DE) and particle swarm optimisation (PSO), are used to evolve single-walled carbon nanotube (SWCNT) / liquid crystal (LC) composites capable of classifying artificial, two-dimensional, binary linear and non-linear separable and merged datasets at low SWCNT concentrations. The difference in search behaviour between the two algorithms is found to affect differently the composite' state during training, which in turn affects the accuracy, consistency and generalisation of evolved solutions. SWCNT/LC processors are also able to scale to complex, real-life classification problems. Crucially, results suggest that problem complexity influences the properties of the processors. For more complex problems, networks of SWCNT structures tend to form within the composite, creating stable devices requiring no configuration voltages to classify data, and with computational capabilities that can be recovered more than several hours after training. A method of programming the dynamic composites is demonstrated, based on the re-application of sequences of configuration voltages which have produced good quality SWCNT/LC classifiers. A second set of investigations aims at exploiting the properties presented by the dynamic nanocomposites, whilst also providing a means for evolved device encapsulation, making their use easier in out-of-the lab applications. Novel composites based on SWCNTs dispersed in one-part UV-cure epoxies are introduced. Results obtained with these composites support their choice for use in subsequent EiM research. A final discussion is concerned with evolving an electro-biological processor and a memristive processor. Overall, the work reported in the thesis suggests that dynamic nanocomposites present a number of unexpected, potentially attractive properties not found in other materials investigated in the context of EiM.

Declaration of Authorship

I, Eleonore Gabrielle Blanche Vissol-Gaudin, declare that this thesis titled, "Evolutionary computation based on nanocomposite training: application to data classification" and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed: Eleonore Gabrielle Blanche Vissol-Gaudin

Date: 28/04/2020

Statement of Copyright

Aknowledgments

I would first like to thank Dr Apostolos Kotsialos for introducing me to the very exciting field of evolution in materio (EiM), for teaching me all there is to know about optimisation and evolutionary algorithms across undergraduate and postgraduate, and for the continuous trans-European support, collaboration and exceptional patience. Very many thanks must also be given to my supervisors, Prof. Dagou Zeze, Dr Chris Groves and Prof. Mike Petty for their invaluable guidance, support and help, without which this PhD would have been much more difficult to complete. Special thanks to Dr Chris Pearson for all the help with material preparation and the lab-based frustration shared on many occasions. Thanks also to Dr Noura Al-Moubayed for the collaboration and the many interesting and motivating discussions. I would also like to thank Dr Kieran Massey for sharing his experience and knowledge of the material and electronics aspects of the field with me, and Maude Pillot and Gabin Gregoire for being really inspiring colleagues to work with. Finally, many thanks to Ian Hutchinson and everyone else in the electronics lab for their kind help.

Je tiens à remercier ma mère pour m'avoir transmis ce besoin de connaissance et cet esprit d'analyse qui m'ont dirigé vers, et permis d'entreprendre, ce travail de recherche, mais aussi pour son enthousiasme, sa motivation et sa lecture attentive et critique de chacun de mes articles, et de ma thèse. Merci aussi a Bernadette, qui m'a donné envie de poursuivre mes études aux Royaumes Unis, et pour son support continu durant les années de licence et de thèse. Thanks to Richard for all the discussions, travels, museum expeditions and books which formed my mind throughout the years, I would have liked to be able to share these experiences with him again after the thesis-writing hyatus.

Merci à Bertille, Jeanne, Kim, Alice, Sophie, Athina et Ferdi pour les années de discussions, de fous-rires et d'amitié. Thanks to William for being there through all the fun and less fun bits of the PhD. Thanks also to Rob and Tom for the mental support throughout 7 years in Durham, via pints, wine and extremely geeky late-night debates. Thanks to Celine, Tim, Nick and Nang for making Le Oyapeyasom an experience that made me want to get into research. Thanks to Hannah, Jen, Rosie, Yasmine, Bella, Al, Jacqueline, Kevin, Suyin, Alexandre, Javier and Enguerrand for their friendship and giving me some concepts in research areas outside Engineering, and thanks to Ben, George and Luis for being great company over the last few months of writing and for all the enthusiastic scientific discussions. I must also thank all friends and family I have not mentioned directly but are always on my mind and who have been incredibly supportive.

Last, but not least, I must thank the Department of Engineering of Durham University for awarding me a scholarship to pursue research in the field of EiM, and providing me with the facilities and support to undertake this research. Thanks must also be given to Hatfield College, its MCR and Trust, which provided me with great facilities, a de-facto family and financial support, respectively. Specifically, I would like to thank Profs Tim Burt, Ann Maclarnon, Anthony Bash and Dr Ellen Crabtree for their support and many interesting discussions, and must also be thanked all the college staff who make daily life and special events so easy and enjoyable.

List of Publications

M. K. Massey, A. Kotsialos, D. Volpati, **E. Vissol-Gaudin**, C. Pearson, L. Bowen, B. Obara, D. A. Zeze, C. Groves, and M. C. Petty, 2016. “Evolution of electronic circuits using carbon nanotube composites”, Scientific reports, 6, p.32197

E. Vissol-Gaudin, A. Kotsialos, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Evolution-in-Materio of Single-Walled Carbon Nanotubes Networks in Liquid Crystal: Investigations with Binary Classification Problems", Submitted to the International Transactions on Evolutionary Computation, IEEE.

E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, “Training a carbon-nanotube/liquid crystal data classifier using evolutionary algorithms”, Unconventional Computation and Natural Computation Conference, Lecture Notes in Computing Sciences, pp. 130–141, 2016.*

E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, “Data classification using carbon-nanotubes and evolutionary algorithms”, International Conference on Parallel Problem Solving from Nature, Lecture Notes in Computing Sciences, pp. 644–654, 2016.**

E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, C. Groves, C. Pearson, D.A. Zeze, and M. C. Petty, “Solving binary classification problems with carbon nanotube/liquid crystal composites and evolutionary algorithms.”, 2017 IEEE Congress on Evolutionary Computation (CEC), pp 1924-1931. IEEE.*

E. Vissol-Gaudin, A. Kotsialos, C. Groves, C. Pearson, D. A. Zeze, and M. C. Petty, “Computing based on material training: application to binary classification problems”, 2017 IEEE International Conference on Rebooting Computing (ICRC), pp. 1–8. IEEE.*

E. Vissol-Gaudin, A. Kotsialos, N. Al-moubayed, C. Groves, C. Pearson, D.A. Zeze, and M.C. Petty, "Confidence Measures for Carbon-Nanotube / Liquid Crystals Classifiers", 2018 IEEE World Congress on Computational Intelligence (WCCI 2018), pp. 1-8. IEEE*

* *paper presented orally*
** *paper presented in poster format*

Conferences and Workshops

E. Vissol-Gaudin, "Training a Carbon-Nanotube / Liquid Crystal Data Classifier Using Evolutionary Algorithms", 2016 Inaugural Mechanics Research Day, Department Of Engineering, Durham University, Durham, UK.***

E. Vissol-Gaudin, A. Kotsialos, M.K. Massey, D.A. Zeze, C. Pearson, C. Groves and M.C. Petty, "Data Classification using Carbon-Nanotubes and Evolutionary Algorithms", Research Day 2016, School of Engineering and Computing Sciences, Durham University, Durham, UK. (3rd Poster Prize)****

E. Vissol-Gaudin, A. Kotsialos, M.K. Massey, D.A. Zeze, C. Pearson, C. Groves and M.C. Petty, "Data Processing in Materio using Evolutionary Algorithms", 2017 Inaugural Castle Conference, Evolution, Durham University ***

E. Vissol-Gaudin, A. Kotsialos, C. Groves, C. Pearson, D.A. Zeze and M.C. Petty, "Computing Using Carbon Nanotubes", Research Day 2017, Department of Engineering, Durham University, Durham, UK. (3rd Oral presentation Prize) ***

E. Vissol-Gaudin, A. Kotsialos, C. Groves, C. Pearson, D.A. Zeze and M.C. Petty, "Solving Binary Classification Problems with Carbon Nanotube / Liquid Crystal Composites and Evolutionary Algorithms", Constructal Law and Physical Geography Workshop, Invited talk, Department of Geography, Durham University. (2017). ***

E. Vissol-Gaudin, A. Kotsialos, C. Pearson, C. Groves, M.C. Petty and D.A. Zeze, "Using carbon-nanotube classifiers to diagnose diseases" 2018 International Workshop and School. Nanostructure for Photonics (NSP2018). St Petersburg, Russia (3rd Poster Prize) ****

*** *Oral presentation*
**** *Poster presentation*

Indicator of Merit

Highlight of the work presented at the 2017 ICRC conference in Washington in IEEE spectrum of January 2018:
S. K. Moore, "4 strange new ways to compute [News]," in IEEE Spectrum, vol. 55, no. 1, pp. 10-11, January 2018.
URL:http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8241695&isnumber=8241685

Contents

List of Figures

List of Tables

List of Abbreviations

AC	**A**nt **C**olony
ADC	**A**nalogue to **D**igital **C**onverter
AL	**A**rtificial **L**earing
ANN	**A**rtificial **N**eural **N**etwork
AFM	**A**tomic **F**orce **M**icroscopy
APSO	**A**daptative **P**article **S**warm **O**ptimisation
BCPs	**B**inary **C**lassification **P**roblems
BPNN	**B**ack-**P**ropagation**N**eural-**N**etwork
CA	**C**ellular **A**utomata
CAiM	**C**ellular **A**utomata in **M**aterio
CGP	**C**artesian**G**enetic **P**rogramming
CI	**C**omputational **I**ntelligence
CMOS	**C**omplementary-**M**etal-**O**xide-**S** emiconductor
CMTNN	**C**o**M**plemen**T**ary **N**eural- **N**etwork
CNFET	**C**arbon **N**anotubes **F**ield textbfEffect**T** ransistors
CNTs	**C**arbon**N**ano **T**ubes
DAC	**D**igital to **A**nalogue **C**onverter
DC	**D**irect **C**urrent
DE	**D**ifferential **E**volution
DMNN	**D**entrite-**M**orphological-**N**eural-**N**etwork
DRAM	**D**ynamic **R**andom **A**ccess **M**emory
EA	**E**volutionary **A**lgorithm
EC	**E**volutionary **C**omputating
EH	**E**volvable **H**ardware
EiM	**E**volution in **M**aterio
EM	**E**volvable **M**otherboard
ES	**E**volutionary **S**trategy
FP	**F**alse **P**ositive
FN	**F**alse **N**egative
FinFET	**F**in **F**ield **E**ffect **T**ransistor
FoM	**F**igure**o**f**M**erit
FPAA	**F**ield-**P**rogrammable-**A**nalog-**A**rray
FPGA	**F**ield-**P**rogrammable-**G**ate-**A**rray
FPMA	**F**ield-**P**rogrammable-**M**atter-**A**rray
FPTA	**F**ield-**P**rogrammable-**T**ransistor-**A**rray
FTM	**F**ilm-**T**hickness-**M**onitor
GA	**G**enetic **A**lgorithm
GAA	**G**ate-**A**ll- **A**round
GP	**G**enetic **P**rogramming
GRNN	**G**eneral-**R**egession**N**eural-**N**etwork
GPSO	**G**lobal **P**article **S**warm **O**ptimisation
IC	**I**ntegrated **C**ircuits
IRDS	**I**nternational **R**oadmap for **D**evices and **S** ystem
KNN	**K**-**N**earest **N**eighbour

LC	**L**iquid **C**rystal
LCD	**L**iquid **C**rystal **D**isplay
LHS	**L**eft **H**and **S**ide
LPSO	**L**ocal **P**article **S**warm **O**ptimisation
LR	**L**ogistic **R**egression
MMC	**M**ammographic-**M**ass- dataset
MOSFET	**M**etal-**O**xide **S**xide **S**emiconductor **F** ield-**E**ffect **T**ransistor
MSE	**M**ean-**S**quarred-**E**rror
MWCNT	**M**ulti-**W**alled **C**arbon-**N**ano **T** ubes
NASCENSE	**NA**no**SC**ale **E**ngineering for **C**omputation using **E**volution
NC	**N**atural **C**omputing
NM	**N**elder- **M**ead
NN	**N**eural- **N**etwork
PC	**P**ersonal **C**omputer
PCB	**P**rinted **C**ircuit **B**oard
PBMA	**P**oly-**B**utyl-**M**etha-**A**crylate
PMMA	**P**oly-**M**ethyl-**M**etha-**A**crylate
POEtic	**P**hylogenetic **O**ntogenetic **E**pigene**tic**
PNN	**P**robabilistic **N**eural-**N**etwork
PSO	**P**article **S**warm **O**ptimisation
RBFNN	**R**adial **B**asic **F**unction**N**eural-**N** etwork
RC	**R**eservoir **C**omputing
RCiM	**R**eservoir **C**omputing in **M**aterio
RHS	**R**ight **H**and **S**ide
RNN	**R**ecurrent **N**eural **N**etwork
RoC	**R**ate **o**f **C**hange
SA	**S**immulated **A**nealing
SD	**S**ecure **D**igital card
SEM	**S**canning **E**lectron **M**icroscopy
SI	**S**warm **I**ntelligence
SIA	**S**warm **I**ntelligence **A**lgorithm
SRAM	**S**tatic **R**andom **A**ccess **M**emory
SSIM	**S**tructural-**S**imilarity-**I**ndex-**M**easure
SWCNT	**S**ingle-**W**alled-**C**arbon-**N**anotube
SWCNT/LC	**S**ingle-**W**alled-**C**arbon-**N**ano**T** ube**L**iquid-**C**rystal**N**anotube
TP	**T**rue **P**ositive
TN	**T**rue **N**egative
UC	**U**nconventional **C**omputing
UCI	**U**niversity of **C**alifornia **I**rvine
USB	**U**niversal **S**erial **B**us
UV	**U**ltra - **V**iolet
X-OR	e**X**clusive- **OR**

Nomenclature

n_1	number of *computation* input	
n_2	number of *configuration* input	
n_3	number of output measurements	
n_4	number of additional problem variables	
$\mathbf{V}^{\mathcal{C}}$	vector of *computation* inputs	
$\mathcal{C}()$	computation outcome, i.e. known class of inputs	
$\mathbf{x}$	vector defining the state of a device	
$\mathbf{V}$	vector of *configuration* inputs	
$\mathbf{M}$	material state	
$\mathbf{R}$	vector of additional problem variables	
$\mathbf{x}'$	only the well defined quantities of $\mathbf{x}$, i.e. not $\mathbf{M}$	
$\mathbf{Y}(\mathbf{M})$	output measurements	
$\mathcal{C}_M()$	computation outcome based on output measurements	
$S_{\mathcal{C}}$	interpretation scheme ('translates' $\mathbf{Y}(\mathbf{M})$ into $\mathcal{C}_M()$)	
T_1 and T_2	error threshold used in the termination criteria	
K	number of input pair (attribute/class) in a dataset	
Λ	maximum number of iterations	
Q	maximum number of verification tests	
N	population size of the EA(s)	
D	number of EA-controlled dimensions ($= n_2 + n_4$)	
λ	iteration index	
l	individual index	
k	computation input/class pair (instance) index	
t	indicates a belonging to the training dataset	
v	indicates a belonging to the verification dataset	
i	verification test index	
*	indicates the best result achieved	
$^-$	indicates an average	
K_t	number of training input/class pairs (instances)	
K_v	number of verification input/class pairs (instances)	
Φ_e^t	training error averaged over K_t	%
$\Phi_e^{v,i}$	verification error averaged over K_v, for one test i	%
$\Phi_e^{t,*}$	best error obtained during training (over all $\lambda \leq \Lambda$)	%
$\Phi_e^{v,*}$	best error obtained during verification tests (over all $i \leq Q$)	%
$\overline{\Phi_e^v}$	error obtained during verification tests averaged over Q	%
$\overline{d_\Phi}$	difference between best training and best verificatin errors	%
$\sigma_{\Phi_e^v}$	standard deviation in verification error across experiments	%
σ_Φ	standard deviation in verification error across verification tests	%

Chapter 1
Introduction

Evolution in materio (EiM) is a field of research within the context of unconventional computing (UC), where evolutionary algorithms (EAs) are used to explore and exploit the properties of materials, with the aim of solving computational problems. EiM has demonstrated the ability to solve a number of computational problems of varying complexity following a different approach to that followed using conventional computers. Since most EiM research has been based on experimental implementations rather than models, only few algorithm / material combinations have been studied extensively. The work presented here introduces new materials and algorithms, based on the hypothesis that a better understanding of the impact of the latter on the former might provide means for optimising EiM, or lead to the discovery of advantages of EiM over conventional computing techniques. In all experiments reported, materials were evolved with the aim of transforming them into devices able to classify data.

1.1 Context

The question of how to create a machine able to compute raises a number of additional questions such as what is computation, to what level is a machine computing, and whether computation implemented in machines can represent, or are a good representation of, the processes occurring in the brain. The first modern attempts at solving these questions include work by Babbage, Lovelace [1], Zuse [2] and Turing [3]. In each case, both a means of representing problems and a machine in which to implement the solver of this problem were proposed. In other words, these early works aimed at automating computation by means of machines, with the aim of replicating the process of

thought. Turing's abstract model of computation first presented in Computable Numbers [4, 5], had three main advantages. It is universal, ie: any Turing computable problem can be solved by a Turing machine. It is material-independent, i.e. it can be realised in any medium presenting two well-defined distinct states, in other words, acting as a transistor. Finally, the computational power of Turing machines (their ability to solve a problem) is, ideally, only limited by the time is takes to flip between the two distinct states and to transfer information between each transistor, if more than one is involved. The computational power is therefore dependent on the size of the hardware and the ability of the user of a Turing machine to transform a problem into its simplest Turing-computable form. This theory of computation and implementation, along with the work undertaken by Von Neumann [6], has enabled the development of the technology that has become ubiquitous in the XXIst century.

The current impact of this technology (personal computers, smartphones, etc) on society can be assessed in terms of global use of computing technologies. It has been estimated in 2017 that across the globe, 46.9% of households have a computer and 48.6% individuals use the internet [7, 8]. In addition to personal use, the number and scale of applications requiring advanced computing technology is ever increasing. A large area of research is focusing on the automation of medical data analysis [9, 10]. Other areas include security [11], smart cities [12–14] and autonomous vehicles [15]. These developments have been made possible by constant increases in computer speed, accuracy and efficiency, resulting primarily from reductions in the size, at constant cost, of the computer's basic element: the silicon-based transistors [16]. As predicted by Moore's 'law' [17] in the 1970s, the last 40 years have seen a near exponential growth in the number of transistors that can be built on a chip, for the same or reduced cost. This has led to important improvements in conventional electronic technologies. The latest metal-oxide-semiconductor-field-effect-transistors (MOSFET), building block of most conventional electronic circuits, have reached 14 nm gate length [18, 19] allowing chips to contain 1.3 billion transistors. Recent publications also reported the successful production of 10 nm [20] and 7 nm [21] technology.

Moore's law however, is generally accepted to be reaching its limits. The density of transistors on a chip is now generally accepted to double every 2.5 years whilst the cost of production is consistently increasing. In addition, the top-down discrete approach

to computation is currently unable to solve efficiently (at relatively low computational cost) certain classes of problems such as those involving large non-linear datasets. Further developments are constrained by the Turing model's three main limitations: whilst a reduction in transistor size increases operational frequency of MOSFETs, and therefore their speed, it also increases power dissipation, which is a direct relation of the number of transistors and the power they consume, thereby limiting the number of transistors than can feasibly be contained within one chip [22]; the fact that the model is material independent means that it does not take into account characteristics specific to the material, resulting in potential loss in efficiency and complexity [23]; finally, a Turing machine can solve any Turing-computable problem, but some problems might not be Turing-computable [24, 25].

In order to identify the areas of research that could by-pass these limitations or provide alternatives to the current technology, international bodies combining academia and industry have been set up. The international technology roadmap for semiconductors (ITRS), which ran from 1998 to 2015, enabled the development of new technology such as Fin-field effect transistors (FinFET) [26] and gate-all-around (GAA) nanowire/nanosheet structures [27] which can replace typical planar silicon transistor to reach 10 nm and 5 nm scales. In this case, a conventional approach was followed in the sense that the aim was to obtain devices with optimised performances but with minimal changes to the production process and high integration in current technology. The international roadmap for devices and systems (IRDS) [28], set up in 2017, follows on the work done by the ITRS but has expanded the focus to include computing frameworks and systems that deviate from the current conventional paradigm of computation.

1.2 Unconventional Computing and Evolution in Materio

A field of research which focuses on unconventional approaches to the finding of alternatives to the current computing framework and technology is unconventional computing (UC). It has been defined as a field that "deals with computing and information processing derived from or implemented in physical, chemical and biological systems "[29].

Neuromorphic, analogue and quantum computing are examples of interesting avenues of research within UC. Reviews of the state of the fields and the associated technology can be found in [30–32]. It must be noted that [30] and [32] date from 2013 and

2010, respectively, and might not, therefore present the latest developments. However, these three areas of research do not constitute the core subject of this Thesis and are only discussed briefly. On the other hand evolvable hardware (EH) and evolution in materio (EiM) are described in greater length as they are directly relevant to the investigations.

UC is a relatively new and inter-disciplinary area of research. Consequently, specific terms employed to describe and define UC concepts in literature vary from author to author. The terminology employed here is the one found to be the most appropriate and relevant to the paradigm discussed by the author of this thesis.

1.2.1 Example of Unconventional Computing

Neuromorphic computing is an example of UC which is currently inspired by, rather than implemented in, biological systems. This alternative framework is based on the current knowledge of information processing performed in the brain. It has already been used to solve technological and industrial problems, but remains in the sphere of research. Neuromorphic operations have been modelled in conventional computers. However, based on the argument that neuromorphic computing could benefit from running on a different type of system architecture, the SpiNNaker project [33] has shown that highly parallel spiking transistors are better suited than conventional transistors for the purpose of running this type of processes. Other examples of computer architecture developed specifically for neuromorphic computing and based on silicon technology are the True North chip [34] and the Loihi research chip [35] developed by IBM and Intel, respectively. Non silicon-based materials have also been proposed as more natural choices to run neuromorphic operations, such as memristors [36, 37]. Indeed, they present the spiking behaviour characteristic to biological neurons. The main advantage of neuromorphic computers is that memory and computation are located in the same place. As a result, the number of components required to perform a given operation is lower, potentially reducing errors arising from data transfer. However, this is done at the expense of speed.

Analogue computing is another example of UC. In this case, a given physical system is modelled using a different physical system. This analogue of the original system follows the same basic working principles (is defined by the same set of equations), but is easier and/or less costly to implement. Problems in the original system can therefore be solved by solving their equivalent system analogue [38]. Analogue computing must not

be confused with analogue electronics, which consists in the use of components such as linear resistors or capacitors. Whilst the latter can be used to model complex dynamical systems [39], they are not the only hardware used in this framework. For example, at the time of the first digital computers, an optical analogue computer was proposed [40]. In this case, the light intensity was considered as a signal analogue, and the response of a photosensitive substract to this intensity was used to 'compute' the two dimension Fourier transform of a function. More recently, a case of optical analogue computing was made in [41], where it is argued that some problems with an optical analogue have the potential of being solved much faster using this paradigm than by simulation in a digital computer. The fact that analogue computing tends to be material dependent can have the advantage of use of all available resources provided by the hardware [31]. However, analogue computers tend to be task-specific and suffer from a lack of consistency or accuracy in the solution obtained, mainly due to noise [31, 41].

Quantum computing is a well-known example of UC which is inspired by, and implemented in, physical systems. The aim is to exploit the quantum mechanical properties inherent in materials at small scales, in order to solve computational problems. This required the development of a new theory of computation, quantum computation, in addition to new devices that can implement this theory. It is difficult to find a proven numerical estimate of the percentage increase in speed and efficiency provided by the use of a quantum computer over a digital one. However, a combination of theoretical proof and experimental results reported in [42] suggest a clear advantage. Quantum computers [43] have been simulated in digital computers [44], but quantum computing-specific hardware has also been successfully developed [45, 46]. Research is still on-going, however, preliminary results suggest that the potential for errors in reading of the quantum bit (qubit) increases exponentially with the number of qubit, thereby constituting this framework's main limitation [32].

1.2.2 Evolvable Hardware

Evolvable hardware (EH) is another example of UC. EH is concerned with using evolutionary algorithms (EAs) to produce circuit designs in flexible hardware architectures.

EAs are generally population-based heuristic search algorithms [47, 48] which follow the working principles of natural systems. They are also iterative and can involve

a degree of stochasticity. Well-known EAs include genetic algorithm (GA), for which a basic iteration is presented in Figure 1.1, evolutionary computing (EC) and differential evolution (DE). Other algorithms include particle swarm optimisation (PSO) and ant colony (AC), which belong to swarm intelligence (SI). All have been studied, developed and applied successfully to varied problems such as travelling salesman, power system efficiency increase, or financial risk [49–51].

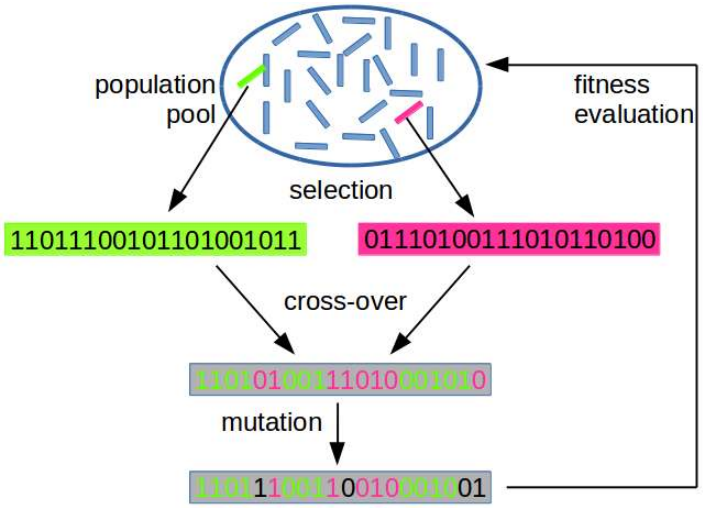


FIGURE 1.1: Simple schematics of the work-flow of a genetic algorithm, with a population of possible solutions to a computation problem, and the fitness-biassed selection, cross-over and mutation operations resulting in a new individual in the population

The concept of EH was first proposed by Thompson following a series of experiments reported in [52–54]. In [52], Thompson attempted to create a tone discriminator circuit out of a field-programmable-gate-array (FPGA) using a genetic algorithm (GA) [54]. Connections between the FPGA's components were controlled by the algorithm which was run for a number of iterations free of constraints such a clock frequency, specific waveforms, etc. It was observed that the resulting circuit was very different from one carefully designed using a conventional approach. In addition, it was observed that the solution selected by the algorithms utilised the components typically considered digital in an analogue manner. In [54], it was suggested that the feedback provided by the iterative nature of the stochastic optimisation interacting with the material allowed the identification of solutions based on the specific FPGA's properties that were unaccounted for during the board's design. In other words, the evolved circuit topologies were influenced by the material making up the hardware components used [23]. The properties of this hardware were explored by the EAs, making it an experiment in intrinsic evolution of materials.

Two EH approaches can be used to tune reconfigurable hardware circuit to perform specific functions [55]. The extrinsic approach uses a model of the hardware and of the properties assigned by design to its components to simulate its behaviour during evolution. The solution obtained is then implemented in the physical device previously modelled. On the other hand, in the intrinsic approach, evolution is performed directly in the material that constitues the hardware. The resulting circuit is obtained by exploiting the physical richness of the materials that compose the electronic components of the flexible hardware.

Different types of hardware have been used or modelled in EH investigations. Many are based on the same principle as the FPGA but with variations on the basic component, such as field-programmable-transistor-arrays (FPTA), field-programmable-analog-arrays (FPAAs) or the POEtic device [56]. Interesting applications of EH include the design of passive filters [57], fault-tolerant systems [58] and data compression [59].

EH has the ability to overcome some limitations of conventional computing such as fault-tolerance, adaptability and automation of novel design production [52, 59, 60]. However, overcoming these limitations has generally been achieved to the detriment of speed and simplicity. As for EAs, the main limitation of EH's is scalability. Investigations have focused on solving this problem (potential solutions have been reported in [61, 62]), but without making EH competitive compared to conventional techniques. In addition, whilst the main objective behind EH was to exploit the ability of EAs to make use of random or potentially unknown properties of hardware, the components used in the different types of FPGAs were produced, assembled and run in a way that minimises variations in behaviour. This is a requirement of the conventional computing framework. However, it constrains the amount of unknown for the EAs to explore.

1.2.3 Evolution in Materio

Arising from the latter discussion, and the results and observations reported by Thompson regarding the intrinsic evolution of FPGAs, a relatively new field has emerged. This field is referred to as evolution in materio (EiM) after Miller and Downing [63], where they presented a new kind of flexible hardware architecture, the field programmable matter array (FPMA) based on a liquid crystal display.

EiM is a field of UC which aims to bring un-configured physically rich materials to a computation-inducing state by exploiting their underlying properties. Contrary to traditional computing with MOSFET technology, where everything is designed, produced and programmed very carefully, EiM uses a bottom up approach where computation is performed by the material without having explicit knowledge of its internal properties. The main difference with EH is that flexible hardware such as FPGAs are replaced by un-configured material systems, favouring exploitation of their physical properties by the search algorithms, more specifically evolutionary algorithms (EAs).

EAs perform an iterative search where the material is configured until it reaches a state where a pre-specified scheme of interaction is uniquely translated as a computation input/output relationship. Configuration of the material is induced by a combination of incident signals which are either controlled by an EA through a combination of hardware and software or independently through the influence of the environment. The physical implementation of EiM is illustrated in Figure 1.2.

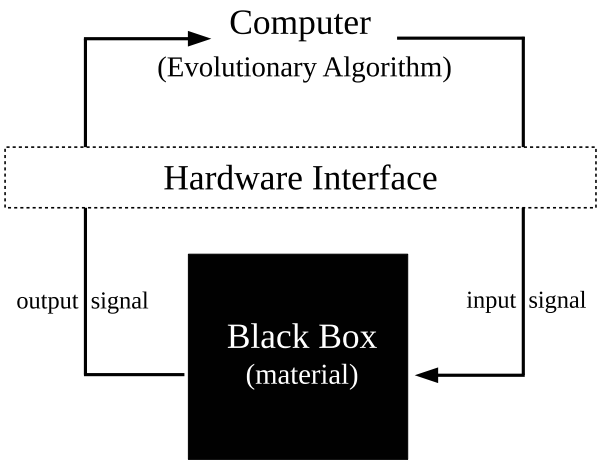


FIGURE 1.2: EiM

A concept similar to EiM can be found in early work of G. Pask [64] concerned with growing an electrochemical ear. Pask began experiments aiming to create a machine able to make use of changes in the material during computation, in order to be more autonomous and flexible. A ferrous sulfate ($FeSO_2$) material was used to create a problem solving circuit, using a reward driven mechanism not unlike some EAs used today [64, 65]. Although the aims were to allow the machines to be self-wiring, self-building and adaptive, the conclusion of the experiment was rather pessimistic, with Pask

describing the process as "lengthy and inefficient, not unlike natural selection". The experiments where abandoned. However, natural selection, and more precisely the process of evolution has been able to create organisms, "biological machines" with a level of complexity far beyond any conventional computing machine created by humans [63].

Contrary to the $FeSO_4$ experiments, EiM has benefited from the developments of digital computers and EAs. In [66], Harding and Miller demonstrated a tone discriminator in liquid crystal (LC), evolved using a genetic algorithm. The same concept was subsequently applied to evolve logic gates [67] and a robot controller [68]. It was observed that the solutions were not very stable, i.e. deteriorating over time, motivating search for the identification of other materials that could be more suited for EiM.

The choice of material was not the only consideration. EiM has broad scope, is interdisciplinary and can be divided in five inter-dependant dimensions visualised in Figure 1.3: (a) the choice of material used (including the physical properties manipulated for obtaining a computation), (b) the hardware or electronics used, (c) the computational problem itself, (d) the formulation of the training problem and (e) optimisation algorithm used for solving it.

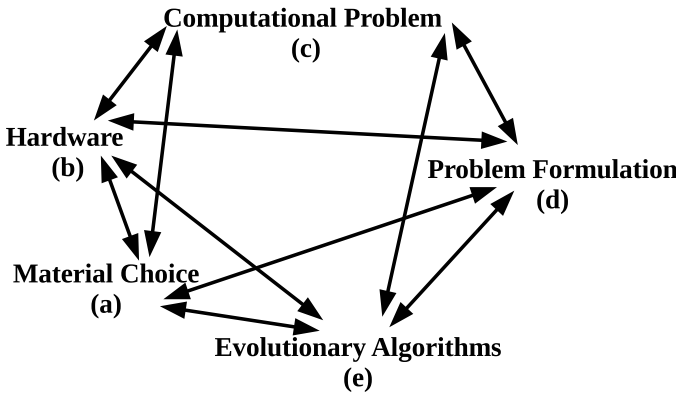


FIGURE 1.3: EiM delineated along five main areas of research

In order to address the different dimensions, a collaboration between five universities was set up, under the name of the NASCENCE project and funded under the European Union's seventh Framework Programme for research (FP7). This project resulted in investigations into different materials, including solid single-walled-carbon-nanotube (SWCNT) / polymer composites [69–71] and dispersions of metallic nanoparticles [72]

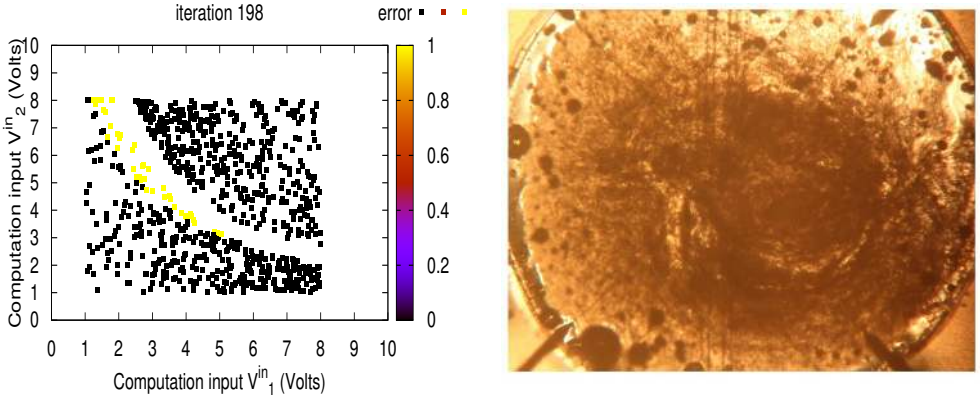
as well as different computational problems[69, 70] and different algorithms [69]. Research aiming at gaining an understanding of the process was also undertaken [73–75], along with the development of a hardware platform for EiM: mecobo [76]. Each dimension of EiM still requires further investigation before it can become widely used as an alternative, or a complement, to conventional computing/electronic technology.

In the most common version of EiM [77], the iterative process is called material training, or evolution, and the post training tests which reuse the optimal evolved solution is called verification. This is a relatively common scheme in learning problems where measurement of generalisation of the solution to new or unseen data is necessary. Training and verification require the selection of two distinct finite sets of data. Both consist of known input/output pairs from the computational problem's domain of definition and range, respectively. The training process requires the repetitive application of inputs sent to the material and measurement of the corresponding response. Measured responses are translated into computation outputs and this allows the definition of an objective function. Specific physical properties of the material are measured for a given EiM implementation. The interpretation scheme of the material's response used for translating these properties into a computation output is pre-specified and fully known before the training process starts.

There are two types of incident signals on the material: computation inputs, which are used to represent the arguments of a computation, and configuration inputs, which are used for changing the material's properties. Modulation of the incident signals is controlled by an optimisation algorithm, which explores the problem's search space. The search space itself is a hybrid of the material's physical state, the hardware used and the subspace spanned by the independent configuration inputs. Hence, the optimisation algorithm aims at configuring the material to a particular state by finding the optimal configuration inputs producing that material state, the response of which can be uniquely translated into a computation.

Following the classical EiM approach, different algorithms have been used to solve a variety of computational problems in a number of materials. The tone discriminator [66], logic gates [67] and evolving robot controllers [68] were evolved in LCs following this approach. In [69, 78, 79] dry composites of SWCNT/ polymer were implemented

as the computational material and its electrical conductance was selected as the manipulated property for solving the problem of calculating Boolean functions with a threshold interpretation scheme; the same material is used in [71] and [80] for solving optimisation problems. In [81–84] the material investigated, along with its ability to solve classification problems, is a SWCNT/LC composite which is in liquid rather than solid state. A sample of this material is presented in Figure 1.4(a) and the classification error that can be obtained with this material on a two-dimensional, binary dataset is illustrated in Figure 1.4(b). Both correspond to the last iteration of the evolutionary process performed during an EiM experiment, when the material is able to successfully classify the majority of the instances from a dataset. The yellow datapoints have an error of 1, i.e. they are incorrectly classified, whilst the black dataspoints have an error of 0, i.e. they are correctly classified. The material and dataset illustrated in the two figures will be further described in the following chapters. They are provided here to give a visual idea of the material → error distribution mapping for a classification problem, and to introduce the images that will be present in the headers and footers from the following chapter. By flipping through the Thesis, the reader shall be able to visualise the evolution process (from iteration 0 to 199) in terms of material and error expected during experiments.



(a) (b)

FIGURE 1.4: For the final step of an EiM process, a microscope photograph of a sample of single-walled-carbon-nanotube / liquid crystal composite used in EiM experiments is presented in (a) and (b) illustrates the resulting sample's classification of instances from a binary dataset.

A current issue with the classical EiM approach has been the difficulty in solving complex computational problems and providing competitive solutions compared with

algorithms run on conventional computers. Each of EiM's dimensions can be investigated to address this issue. Modifying the problem formulation (fig. 1.3 (d)) is one of the possible avenue. An example is the combination of the reservoir computing (RC) framework and EiM [85, 86].

RC was first developed to process outputs of recurrent neural networks (RNN) [87]. Investigations in the field of RC suggest that it is well suited to process outputs from many dynamical systems [88, 89]. The framework has therefore been implemented in various real and simulated media [90, 91] Motivations behind the reservoir computing in materio (RCiM) approach stem from the similarities between EiM and RC's training process [88, 92, 93]. They also stem from the fact that both the typical systems contained in reservoirs and the materials evolved in EiM are complex dynamical systems. Results reported in [94–96] suggest that the RCiM framework can be more suited for the solving of computational problems in solid carbon nanotube-based composites than the classical EiM implementation and in some cases compete with algorithms implemented *in silico*.

The type of material used in EiM remains another subject for further investigations [96]. Without the full understanding of the interactions occurring within the materials during training, it is difficult to prove that one material would be better than another within the context of EiM. This leaves the possibility that a material other than SWCNT/polymer sample, could lead to better computational performance [97]. The field of EiM could also benefit from exploring the impact of processes underlying the evolution of the materials into devices able to compute information.

1.3 Research Hypothesis and Thesis Structure

The research hypothesis is that a dynamic state in SWCNT-based composites can provide an extra layer of complexity as compared to solid SWCNT-based composites and that this complexity can induce unforeseen advantages to the evolved devices. Investigations aim at furthering the current understanding of the interaction between training implementation and material within the context of evolution in materio. If an understanding is gained, it becomes possible to find new properties in liquid devices evolved with EiM: reconfigurability, material programming and material memory.

In order to verify the hypothesis, and follow the aim of the investigations, the Thesis is structured as follows:

Chapter 2, discusses the different materials proposed as potentially attractive for EiM and details the process used to fabricate those chosen in the investigations, along with their electrical characteristics.

Chapter 3 presents a detailed mathematical formulation of the material training and verification problem, along with a description of the EAs used in experiments and the hardware implementation.

Chapter 4 reports the first results obtained when training liquid SWCNT/LC composites to solve synthetic binary computational problems, along with a discussion regarding the importance of the SWCNTs in the composite and whether an optimal concentration of nanotube exists. Different EAs are compared and the impact of their search behaviour on the material state is reported. Finally, implementation parameters are varied in order to study the dependence of the solutions on the choice of their value.

Chapter 5 reports a discussion regarding the concepts of reproducibility, memory, programmability and material retraining of the nanotube / liquid-crystal based composites. A new measure of result confidence is also introduced.

Chapter 6 reports the investigations undertaken in SWCNT-epoxy composites. The focus lies on the potential advantages presented by this material both liquid (dynamic) and solid (static) SWCNT-based composites due to the possibility of finding solutions when the material is in a liquid state, before curing it, effectively encapsulating the solution. Solution stability and deterioration due to the curing process are reported. As for other nanotube-based materials, the effect of concentration is discussed.

Chapter 7 introduces more complex classification problems. Comparisons between results obtained with SWCNT/LCs are compared with those obtained with solid SWCNT-based devices and *in silico* classifiers, i.e. classifiers obtained by algorithms running on silicon-based technology.

Chapter 8 reports preliminary results obtained with microtubules trained to perform data classification and memristors trained to solve Boolean functions.

Finally, Chapter 9 summarises the work presented in this Thesis, from which conclusions are drawn and avenues for future research proposed.

Bibliography

[1] L. F. Menabrea and A. Lovelace, “Sketch of the analytical engine invented by charles babbage,” 1842.

[2] R. Rojas, "Babbage meets zuse: A minimal mechanical computer," in *International Conference on Unconventional Computation and Natural Computation*, pp. 25–34, Springer, 2016.

[3] A. M. Turing, "Computing machinery and intelligence," in *Parsing the Turing Test*, pp. 23–65, Springer, 2009.

[4] A. M. Turing, "On computable numbers with an application to the entscheidungs problem," *Proceedings of the London mathematical society*, vol. 2, no. 1, pp. 230–265, 1937.

[5] A. M. Turing, "On computable numbers with an application to the entscheidungs problem. a correction," *Proceedings of the London mathematical society*, vol. 2, no. 1, pp. 544–546, 1938.

[6] W. Aspray, *John von Neumann and the origins of modern computing*. Mit Press, 1990.

[7] Various, "Measuring the information society report executive summary 2," 2018.

[8] Various, "Measuring the information society report vol. 2," 2018.

[9] V. S. Pendyala and S. Figueira, "Automated medical diagnosis from clinical data," in *2017 IEEE Third International Conference on Big Data Computing Service and Applications (BigDataService)*, pp. 185–190, IEEE, 2017.

[10] A. Skarysz, Y. Alkhalifah, K. Darnley, M. Eddleston, Y. Hu, D. B. McLaren, W. H. Nailon, D. Salman, M. Sykora, C. P. Thomas, and A. Soltoggio, "Convolutional neural networks for automated targeted analysis of raw gas chromatography-mass spectrometry data," in *2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, IEEE, 2018.

[11] S. Akcay, M. E. Kundegorski, C. G. Willcocks, and T. P. Breckon, "Using deep convolutional neural network architectures for object classification and detection within x-ray baggage security imagery," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 9, pp. 2203–2215, 2018.

[12] G. McNeill, J. Bright, and S. A. Hale, "Estimating local commuting patterns from geolocated twitter data," *EPJ Data Science*, vol. 6, no. 1, p. 24, 2017.

[13] B. Cheng, S. Longo, F. Cirillo, M. Bauer, and E. Kovacs, "Building a big data platform for smart cities: Experience and lessons from santander," in *2015 IEEE International Congress on Big Data*, pp. 592–599, IEEE, 2015.

[14] A. Camero, J. Toutouh, D. H. Stolfi, and E. Alba, "Evolutionary deep learning for car park occupancy prediction in smart cities," in *International Conference on Learning and Intelligent Optimization*, pp. 386–401, Springer, 2018.

[15] W. Zhang, C. Sun, T. Breckon, and N. Alshammari, "Discrete curvature representations for noise robust image corner detection," *IEEE Transactions on Image Processing*, 2019.

[16] M. M. Waldrop, "More than moore," *Nature*, vol. 530, no. 7589, pp. 144–148, 2016.

[17] G. E. Moore, "Progress in digital integrated electronics," in *Electron Devices Meeting*, vol. 21, pp. 11–13, 1975.

[18] C. H. Jan, F. Al-amoody, H. Y. Chang, T. Chang, Y. W. Chen, N. Dias, W. Hafez, D. Ingerly, M. Jang, E. Karl, S. K. Y. Shi, K. Komeyli, H. Kilambi, A. Kumar, K. Byon, C. G. Lee, J. Lee, T. Leo, P. C. Liu, N. Nidhi, R. Olac-vaw, C. Petersburg, K. Phoa, C. Prasad, C. Quincy, R. Ramaswamy, T. Rana, L. Rockford, A. Subramaniam, C. Tsai, P. Vandervoorn, L. Yang, A. Zainuddin, and P. Bai, "A 14 nm soc platform technology featuring 2nd generation tri-gate transistors, 70 nm gate pitch, 52 nm metal pitch, and 0.0499 um2 sram cells, optimized for low power, high performance and high density soc products," *VLSI Technology (VLSI Technology), 2015 Symposium on*, pp. T12–T13, June 2015.

[19] S. Novak, C. Parker, D. Becher, M. Liu, M. Agostinelli, M. Chahal, P. Packan, P. Nayak, S. Ramey, and S. Natarajan, "Transistor aging and reliability in 14nm tri-gate technology," *2015 IEEE International Reliability Physics Symposium*, pp. 2F.2.1–2F.2.5, April 2015.

[20] R. Courtland, "Moore's law's next step: 10 nanometers," *IEEE Spectrum*, vol. 54, no. 1, pp. 52–53, 2017.

[21] P.-L. Yang, T. B. Hook, P. J. Oldiges, and B. B. Doris, "Vertical slit fet at 7-nm node and beyond," *IEEE Transactions on Electron Devices*, vol. 63, no. 8, pp. 3327–3334, 2016.

[22] R. Puers, L. Baldi, M. Van de Voorde, and S. E. Van Nooten, *Nanoelectronics: Materials, Devices, Applications, 2 Volumes*. John Wiley & Sons, 2017.

[23] A. Thompson and P. Layzell, "Analysis of unconventional evolved electronics," *Communications of the ACM*, vol. 42, no. 4, pp. 71–79, 1999.

[24] S. G. Akl, "Nonuniversality in computation: fifteen misconceptions rectified," in *Advances in Unconventional Computing*, pp. 1–30, Springer, 2017.

[25] B. J. MacLennan, "Physical and formal aspects of computation: Exploiting physics for computation and exploiting computation for physical purposes," in *Advances in Unconventional Computing*, pp. 117–140, Springer, 2017.

[26] C. Auth, A. Aliyarukunju, M. Asoro, D. Bergstrom, V. Bhagwat, J. Birdsall, N. Bisnik, M. Buehler, V. Chikarmane, G. Ding, *et al.*, "A 10nm high performance and low-power cmos technology featuring 3 rd generation FinFET transistors, self-aligned quad patterning, contact over active gate and cobalt local interconnects," in *2017 IEEE International Electron Devices Meeting (IEDM)*, pp. 29–1, IEEE, 2017.

[27] N. Loubet, T. Hook, P. Montanini, C.-W. Yeung, S. Kanakasabapathy, M. Guillom, T. Yamashita, J. Zhang, X. Miao, J. Wang, *et al.*, "Stacked nanosheet gate-all-around transistor to enable scaling beyond finfet," in *2017 Symposium on VLSI Technology*, pp. T230–T231, IEEE, 2017.

[28] Various, "Irds^{TM} 2016 edition white papers," 2017.

[29] A. Adamatzky, *Unconventional Computing, Introduction to*, pp. 9705–9706. New York, NY: Springer New York, 2009.

[30] J. Hasler and H. B. Marr, "Finding a roadmap to achieve large neuromorphic hardware systems," *Frontiers in neuroscience*, vol. 7, p. 118, 2013.

[31] O. Bournez and A. Pouly, "A survey on analog models of computation," *arXiv preprint arXiv:1805.05729*, 2018.

[32] T. D. Ladd, F. Jelezko, R. Laflamme, Y. Nakamura, C. Monroe, and J. L. O'Brien, "Quantum computers," *Nature*, vol. 464, no. 7285, p. 45, 2010.

[33] S. Furber and S. Temple, "Neural systems engineering," in *Computational intelligence: A compendium*, pp. 763–796, Springer, 2008.

[34] F. Akopyan, J. Sawada, A. Cassidy, R. Alvarez-Icaza, J. Arthur, P. Merolla, N. Imam, Y. Nakamura, P. Datta, G.-J. Nam, *et al.*, "Truenorth: Design and tool flow of a 65 mw 1 million neuron programmable neurosynaptic chip," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 34, no. 10, pp. 1537–1557, 2015.

[35] M. Davies, N. Srinivasa, T.-H. Lin, G. Chinya, Y. Cao, S. H. Choday, G. Dimou, P. Joshi, N. Imam, S. Jain, *et al.*, "Loihi: A neuromorphic manycore processor with on-chip learning," *IEEE Micro*, vol. 38, no. 1, pp. 82–99, 2018.

[36] L. Xu, C. Li, and L. Chen, "Analog memristor based neuromorphic crossbar circuit for image recognition," in *Intelligent Control and Information Processing (ICICIP), 2015 Sixth International Conference on*, pp. 155–160, 2015.

[37] E. M. Gale, "Neuromorphic computation with spiking memristors: habituation, experimental instantiation of logic gates and a novel sequence-sensitive perceptron model," *Faraday discussions*, vol. 213, pp. 521–551, 2019.

[38] C. Beebe, "Model-based computation," in *International Conference on Unconventional Computation and Natural Computation*, pp. 75–86, Springer, 2016.

[39] E. Tamaseviciute, A. Tamasevicius, G. Mykolaitis, S. Bumeliene, and E. Lindberg, "Analogue electrical circuit for simulation of the duffing-holmes equation," *Nonlinear Analysis: Modelling and Control*, vol. 13, no. 2, pp. 241–252, 2008.

[40] B. J. Howell, "Optical analog computers," *JOSA*, vol. 49, no. 10, pp. 1012–1021, 1959.

[41] D. R. Solli and B. Jalali, "Analog optical computing," *Nature Photonics*, vol. 9, no. 11, p. 704, 2015.

[42] S. Bravyi, D. Gosset, and R. Koenig, "Quantum advantage with shallow circuits," *Science*, vol. 362, no. 6412, pp. 308–311, 2018.

[43] R. P. Feynman, "Quantum mechanical computers," *Foundations of Physics*, vol. 16, pp. 507–531, 1986.

[44] T. Jones, A. Brown, I. Bush, and S. Benjamin, "Quest and high performance simulation of quantum computers," *arXiv preprint arXiv:1802.08032*, 2018.

[45] IBM Q experience. https://www.research.ibm.com/ibm-q/technology/devices/#ibmqx2, 2018.

[46] U. Alvarez-Rodriguez, M. Sanz, L. Lamata, and E. Solano, "Quantum artificial life in an ibm quantum computer," *Scientific reports*, vol. 8, no. 1, pp. 1–9, 2018.

[47] L. N. De Castro, *Fundamentals of natural computing: basic concepts, algorithms, and applications*. CRC Press, 2006.

[48] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[49] C. Segura, S. Botello Rionda, A. Hernández Aguirre, and S. I. Valdez Peña, "A novel diversity-based evolutionary algorithm for the traveling salesman problem," *Proceedings of the 201xf 5 Annual Conference on Genetic and Evolutionary Computation*, pp. 489–496, 2015.

[50] E. Yang, A. T. Erdogan, T. Arslan, and N. Barton, "An improved particle swarm optimization algorithm for power-efficient wireless sensor networks," *Bio-inspired, Learning, and Intelligent Systems for Security, 2007. BLISS 2007. ECSIS Symposium on*, pp. 76–82, Aug 2007.

[51] G. Kendall and Y. Su, "A particle swarm optimisation approach in the construction of optimal risky portfolios.," in *Artificial Intelligence and Applications*, vol. 453, pp. 140–145, 2005.

[52] A. Thompson, "Evolving fault tolerant systems," *1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications (GALESIA)*, vol. 1995, no. 414, pp. 524–529, 1995.

[53] A. Thompson, "Silicon evolution," *Proceedings of the First Annual Conference on Genetic Programming*, vol. 1996, pp. 444–452, 1996.

[54] A. Thompson, "An evolved circuit, intrinsic in silicon, entwined with physics," *Evolvable systems: from biology to hardware*, pp. 390–405, 1996.

[55] L. Sekanina, "Evolvable hardware: From applications to implications for the theory of computation," *Unconventional Computation: 8th International Conference, UC 2009, Proceedings*, pp. 24–36, 2009.

[56] P. C. Haddow and A. M. Tyrrell, "Challenges of evolvable hardware: Past, present and the path to a promising future," *Genetic Programming and Evolvable Machines*, vol. 12, pp. 183–215, 2011.

[57] A. Stoica, D. Keymeulen, R. Zebulum, A. Thakoor, T. Daud, Y. Klimeck, R. Tawel, and V. Duong, "Evolution of analog circuits on field programmable transistor arrays," in *Evolvable Hardware, 2000. Proceedings. The Second NASA/DoD Workshop on*, pp. 99–108, 2000.

[58] W. Barker, D. M. Halliday, Y. Thoma, E. Sanchez, G. Tempesti, and A. M. Tyrrell, "Fault tolerance using dynamic reconfiguration on the poetic tissue," *Evolutionary Computation, IEEE Transactions on*, vol. 11, no. 5, pp. 666–684, 2007.

[59] T. Higuchi, M. Iwata, D. Keymeulen, H. Sakanashi, M. Murakawa, I. Kajitani, E. Takahashi, K. Toda, M. Salami, N. Kajihara, and N. Otsu, "Real-world applications of analog and digital evolvable hardware," *IEEE Transactions on Evolutionary Computation*, vol. 3, no. 3, pp. 220–234, 1999.

[60] P. Cariani, "To evolve an ear. Epistemological implications of gordon pask's electrochemical devices," *Systems Research*, vol. 10, no. 3, pp. 19–33, 1993.

[61] T. Kuyucu, M. A. Trefzer, J. F. Miller, and A. M. Tyrrell, "Task decomposition and evolvability in intrinsic evolvable hardware," *2009 IEEE Congress on Evolutionary Computation, CEC 2009*, pp. 2281–2287, 2009.

[62] T. Gordon and P. Bentley, "Towards development in evolvable hardware," *Proceedings 2002 NASA/DoD Conference on Evolvable Hardware*, 2002.

[63] J. F. Miller and K. Downing, "Evolution in materio: Looking beyond the silicon box," *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on*, pp. 167–176, 2002.

[64] J. Bird and E. Di Paolo, "Gordon Pask and His Maverick Machines," *The Mechanical Mind in History*, pp. 185–211, 2008.

[65] C. Lambert, T. Kalganova, and E. Stomeo, "FPGA-based systems for evolvable hardware," *Proceedings of World Academy of Science Engineering and Technology*, vol. 12, no. 12, 2009.

[66] S. L. Harding and J. F. Miller, "Evolution in materio: A tone discriminator in liquid crystal," *Congress on Evolutionary Computation, 2004. CEC2004.*, vol. 2, pp. 1800–1807, 2004.

[67] S. L. Harding and J. F. Miller, "Evolution in materio: Evolving logic gates in liquid crystal," *Proc. Eur. Conf. Artif. Life (ECAL 2005), Workshop on Unconventional Computing: From cellular automata to wetware*, pp. 133–149, 2005.

[68] S. L. Harding and J. F. Miller, "Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal," *Evolvable Systems: From Biology to Hardware*, pp. 155–164, 2005.

[69] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes.," *International Journal of Unconventional Computing*, vol. 10, no. 5-6, pp. 473–497, 2014.

[70] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.

[71] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving function optimization problems using materials," in *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pp. 1–8, IEEE, 2014.

[72] S. Bose, C. P. Lawrence, Z. Liu, K. Makarenko, R. M. van Damme, H. J. Broersma, and W. G. van der Wiel, "Evolution of a designless nanoparticle network into reconfigurable boolean logic," *Nature nanotechnology*, vol. 10, no. 12, p. 1048, 2015.

[73] D. Laketić, G. Tufte, S. Nichele, and O. R. Lykkebø, "Bringing colours to the black box–a novel approach to explaining materials for evolution-in-materio," in *Proceedings of 7th International Conference on Future Computational Technologies and Applications. XPS Press*, 2015.

[74] S. Nichele, D. Laketic, and G. Tufte, "Is there chaos in blobs of carbon nanotubes used to perform computation?," *7th International Conference on Future Comp. Tech. and Applications, IN PRESS*, 2015.

[75] D. Laketić and G. Tufte, "A hierarchical view on evolution-in-materio computations," in *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–10, IEEE, 2016.

[76] O. R. Lykkebø, S. Harding, G. Tufte, and J. F. Miller, "Mecobo: A hardware and software platform for in materio evolution," in *International Conference on Unconventional Computation and Natural Computation*, pp. 267–279, Springer, 2014.

[77] J. F. Miller, S. L. Harding, and G. Tufte, "Evolution-in-materio: evolving computation in materials," *Evolutionary Intelligence*, vol. 7, no. 1, pp. 49–67, 2014.

[78] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[79] F. Qaiser, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, and M. C. Petty, "Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation," in *IEEE Congress on Evolutionary Computation (CEC)*, pp. 5255–5261, IEEE, 2016.

[80] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving bin packing problems using materials," *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 38–45, 2014.

[81] M. K. Massey, A. Kotsialos, D. Volpati, E. Vissol-Gaudin, C. Pearson, L. Bowen, B. Obara, D. A. Zeze, C. Groves, and M. C. Petty, "Evolution of electronic circuits using carbon nanotube composites," *Scientific reports*, vol. 6, 2016.

[82] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Training a carbon-nanotube/liquid crystal data classifier using evolutionary algorithms," *Unconventional Computation and Natural Computation Conference: 15th International Conference, UCNC 2016*, pp. 130–141, 2016.

[83] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Data classification using carbon-nanotubes and evolutionary algorithms," *International Conference on Parallel Problem Solving from Nature*, pp. 644–654, 2016.

[84] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, C. Groves, C. Pearson, D. A. Zeze, and M. C. Petty, "Solving binary classification problems with carbon nanotube/liquid crystal composites and evolutionary algorithms.," *IEEE Congress on Evolutionary Computation (CEC)*, 2017.

[85] M. Dale, J. F. Miller, and S. Stepney, "Reservoir computing as a model for in-materio computing," in *Advances in Unconventional Computing*, pp. 533–571, Springer, 2017.

[86] Z. Konkoli, S. Nichele, M. Dale, and S. Stepney, "Reservoir computing with computational matter," in *Computational Matter*, pp. 269–293, Springer, 2018.

[87] L. N. de Castro, "Fundamentals of natural computing: an overview," *Physics of Life Reviews*, vol. 4, no. 1, pp. 1 – 36, 2007.

[88] M. Lukoševičius and H. Jaeger, "Reservoir computing approaches to recurrent neural network training," *Computer Science Review*, vol. 3, no. 3, pp. 127–149, 2009.

[89] S. Scardapane, J. B. Butcher, F. M. Bianchi, and Z. K. Malik, "Advances in biologically inspired reservoir computing," *Cognitive Computation*, vol. 9, no. 3, pp. 295–296, 2017.

[90] F. Duport, A. Smerieri, Y. Paquot, B. Schneider, J. Dambre, B. Schrauwen, M. Haeltermann, and S. Massar, "Recent advances in optical reservoir computing," in *2013 IEEE International Symposium on Circuits and Systems (ISCAS2013)*, pp. 333–336, IEEE, 2013.

[91] K. Vandoorne, M. Fiers, T. Van Vaerenbergh, D. Verstraeten, B. Schrauwen, J. Dambre, and P. Bienstman, "Advances in photonic reservoir computing on an integrated platform," in *2011 13th International Conference on Transparent Optical Networks*, pp. 1–4, IEEE, 2011.

[92] A. Goudarzi, M. R. Lakin, and D. Stefanovic, "Reservoir computing approach to robust computation using unreliable nanoscale networks," pp. 164–176, Springer, 2014.

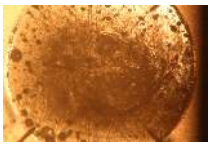
[93] B. Schrauwen, D. Verstraeten, and J. Van Campenhout, "An overview of reservoir computing: theory, applications and implementations," in *Proceedings of the 15th European Symposium on Artificial Neural Networks*, pp. 471–482, 2007.

[94] D. Matthew, J. F. Miller, S. Stepney, and M. A. Trefzer, "Evolving carbon nanotube reservoir computers," *Unconventional Computation and Natural Computation: 15th International Conference, UCNC 2016*, pp. 49–61, 2016.

[95] M. Dale, S. Stepney, J. F. Miller, and M. Trefzer, "Reservoir computing in materio: A computational framework for in materio computing," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2178–2185, IEEE, 2017.

[96] M. Dale, *Reservoir Computing In-Materio (PhD Thesis)*. University of York, 2018.

[97] S. L. Harding, J. Neil, K. Zauner, and J. Miller, "A Framework for the Automatic Identification and Extraction of Computation from Materials," *Computer*, 2008.

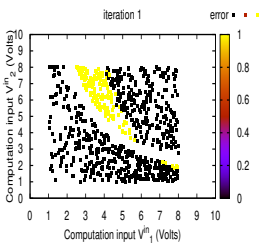


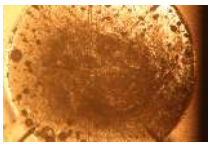
Chapter 2
Materials

2.1 General Characteristics

Evolution in materio (EiM) is a field of unconventional computing (UC) where Evolutionary Algorithms (EAs) are used to explore and exploit the properties of materials, with the aim of transforming them into computing devices. The principles of, and motivations behind, EiM investigations have been outlined in Chapter 1. This field of research provides an attractive framework to study the potential for materials other than silicon to be used in future electronic systems. But which material to choose? Pancomputationalism argues that any physical system is computing [1], leading to very interesting debates regarding whether all systems compute, and if so, how [2, 3].

However, testing all existing materials would be impractical, as not all physical systems can easily be modelled or directly transformed into useful computational devices. Thus, with the aim of narrowing down investigations, a number of characteristics have been identified to select attractive materials for EiM. Since EiM is a field inherently based on experimental research, where implementation constraints must be taken into consideration, it is argued in [4] that the most necessary requirement directing the choice of material used in EiM experiments is an ability to transform input signals into *measurable* outputs.



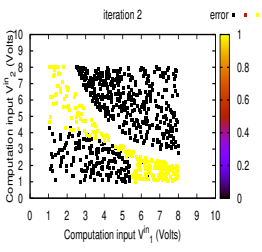


Further discussion [5], based on Thompson's field-programmable-gate-array (FPGA) experiments [6] (Chapter 1), suggests materials need not be configured, such as conventional electronic components, to be used in EiM. It is also suggested that EAs might be able to explore and exploit unconfigured materials better than conventional electronic components, as the properties of the materials have not been constrained to perform specific tasks, thereby potentially reducing the available search space.

Finally, it is suggested that materials presenting a non-linear relationship between inputs and outputs are good candidates for EiM investigations [4, 7]. This feature increases the space of possible solutions that an algorithm can explore in the search for an optimum result to computational problems. The level of complexity required for materials to be used effectively in EiM experiments is discussed in [8] with the conclusion that it might not be necessary for the material to present highly complex electrical and mechanical characteristics for complex problems to be solved. Instead, the suitability of a material, and the level of configuration they need in order to find an optimal solution, will depend on the *in materio* computation it is trained to perform.

A number of investigations have focused on formulating a framework to identify best suited materials for EiM [9] and understanding what happens at the physical level [10–13]. However, the understanding of the relationship between material properties and algorithm's efficiency requires further study. Here, the efficiency is defined in terms of the size of the population used by the algorithm and the number of iterations required before a solution is achieved during training, compared to the accuracy obtained as a result of this training, i.e. closeness to an optimal solution. In the absence of the information that the understanding of the relationship between material properties and algorithm's efficiency would provide, the choice of material was based on empirical results and the materials were treated as black boxes.

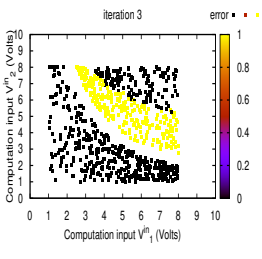
Biological and non-biological materials have been used in EiM and related studies. Examples of biological materials include bacterial consortia [14] and slime moulds [15]. These materials can be tuned to solve specific problems, such as in [16], where a physarium polycephalum was grown into a model of the Tokyo railway network, with the aim of demonstrating the ability of the method to solve transport network design optimisation problems. These two types of biological materials can be viewed as dynamical

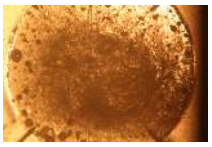


systems, with continuous and stochastic non-linear input/output behaviour. This characteristic gives them the potential to be evolved using the EiM framework in order to produce alternatives to conventional computing hardware. The main limitation to using biological media for EiM is discussed in [17], where it is argued that the natural evolution (as opposed to artificial) to which they have been subjected, results in a bias towards performing tasks that may be unknown or ill-understood. Subsequent evolution using artificial methods to modify the media's behaviour would therefore be less likely to compute information efficiently and accurately.

On the other hand, non-biological media are generally better understood. They present varying degrees of complexity and have not been tuned by natural evolution to perform specific tasks. Liquid crystals (LCs) are an example of non-biological media studied within the EiM framework. LCs from a display screen have been used as the material part of EiM for evolving a robot controller [18], a tone discrimination device [19] and logic gates [20]. In [21–23] a solid composite of Single-Walled Carbon Nanotubes (SWCNT) dispersed in a polymer was used as the computational material; its electrical conductivity was used as the manipulated property for solving the problem of calculating Boolean functions using a threshold interpretation scheme; the same material is used in [24] and [25] for solving optimisation problems.

This chapter first describes the apparatus used to characterise the properties of the materials used in the investigations reported in this work. Four are non-biological, three of which are carbon nanotube composites and the fourth is a memristive device. A biological substrate, microtubules extracted from bovine brains, has also been used. The rationale behind the use of the three different SWCNT-based composites and a resistor array designed as a control sample are detailed in the subsequent discussion, along with the preparation procedure and characteristics. Discussions related to the preparation and the electrical characteristics of the memristive device and the microtubules will be provided in a latter chapter, along with the results regarding their computational response. A summary of the characteristics the SWCNT-based composites selected is presented in the last section of this chapter.





2.2 Gold Micro-Electrode Arrays

2.2.1 Design

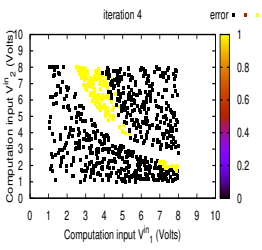
Three different electrode designs have been used, depending on the material and whether it is to be characterised or evolved. These are part of the designs developed during the NASCENCE project and discussed in [4, 26].

The first design, presented in Figure 2.1(a), is a microelectrode array consisting of sixteen electrodes patterned on a glass microscope slide. The external contacts' dimensions are constrained by the edge connectors used in the experimental set up. Each of these contacts are 2.5 mm wide with 1.5 mm separation in between. In the center of the slide, the material contacts are 50 μm with 100 μm pitch. The main function of this electrode array is to provide a means of interacting with the material during EiM experiments. The number of electrodes is a constraint that directs the choice of tasks to be investigated as well as the algorithms' parameters. For example, if a problem is defined by a number of attributes larger than, or very close to, the number of electrodes, it will not be possible to assign one attribute, configuration input, and output per electrode. This will make the problem more difficult to solve than if the number of attributes allows at least one configuration input and one output to each be assigned to an electrode. The design therefore attempts to maximise the number of electrodes, whilst fitting physical constraints presented by experimental hardware and material.

The two other designs of electrode arrays, presented in Figure 2.1(b) have been produced in order to measure the electrical characteristics of the different materials. One array consists in a pair of electrodes 1 mm wide, separated by a 25 μm gap and deposited on 50.8 mm (2 inches) glass wafers. The second consists of four electrodes of 5 μm width and 50 μm central gap deposited on silicon wafers, with a 90° angle between each other.

2.2.2 Fabrication

Wafers and microscope slides were first cleaned with propane-2-ol, acetone, Decon 90 and water following a process detailed in Appendix A. Each electrode array was subsequently patterned upon the respective wafer or slide, using etch-back photolithography, as described below.



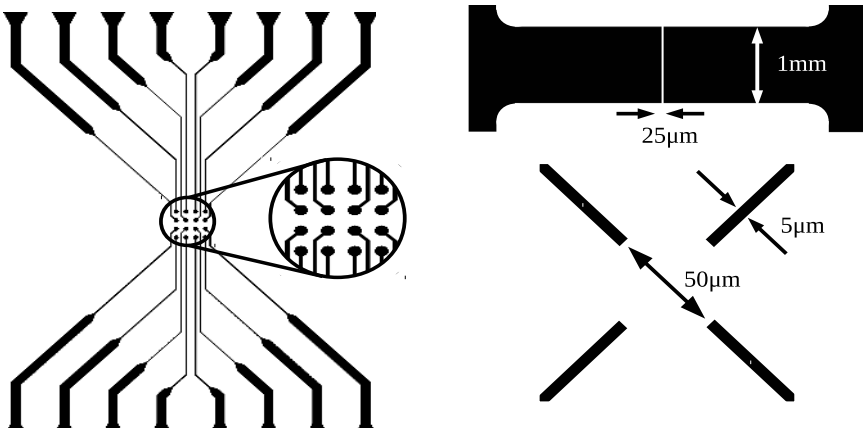


FIGURE 2.1: Electrode pattern for (a) Evolution in Materio experiments and (b) top and bottom, in-plane electrical characterisation of materials for EiM (not to scale).

Thermal evaporation was used to deposit 10 nm chromium (Cr) layer, followed by a 100 nm gold (Au) uniform layers on the slides or wafers' surface. The evaporation was performed by Edwards 306 thermal evaporator in a high vacuum environment (<10^5 mbar). An Edwards RV12 rotary pump backing an Edwards E04K diffusion pump was used to achieve the high vacuum. A quartz crystal microbalance was connected to an Edwards film thickness monitor (FTM7) to monitor deposition rate and film thickness. Slides and wafers were then spin-coated with a layer of SPR350 photoresist and subsequently heated on a hot plate. The spin-coater used for the thin-film deposition was a Laurell Technologies WS-400A-6NPP-LITE.

Following the spin-coating, a mask of the microelectrode array pattern was positioned upon the slides/wafers, which were exposed to a high intensity UV light. An EVG620 Mask Aligner was used in this part of the fabrication to achieve the high accuracy patterning. Following their exposure, samples were left in developing solutions, before being etched to remove Au, Cr and the remaining photoresist.

Each microscope slide can be patterned with two of the micro-electrode arrays presented in Figure 2.1(a), each having sixteen terminals. Up to eight slides can be placed in the thermal evaporator. This means that, assuming perfectly uniform deposition of Cr and Au in the evaporator, up to sixteen microelectrode arrays can be produced in the same batch, all having Cr/Au layers with the same characteristics.

In the case of the electrodes used for electrical characterisation of the materials, seventeen of the two-terminal arrays (Figure 2.1(b) top) can be patterned on each wafer. This number is reduced to seven per wafer for the four terminal arrays (Figure 2.1(b)

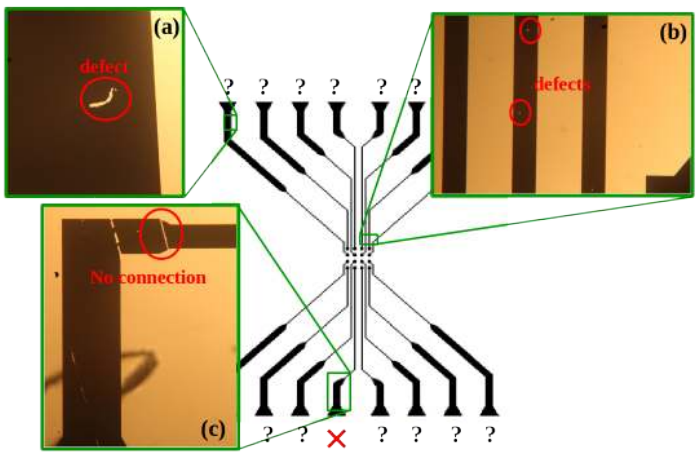
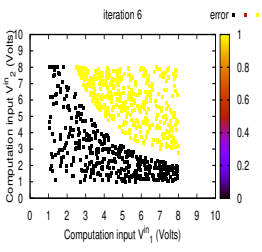


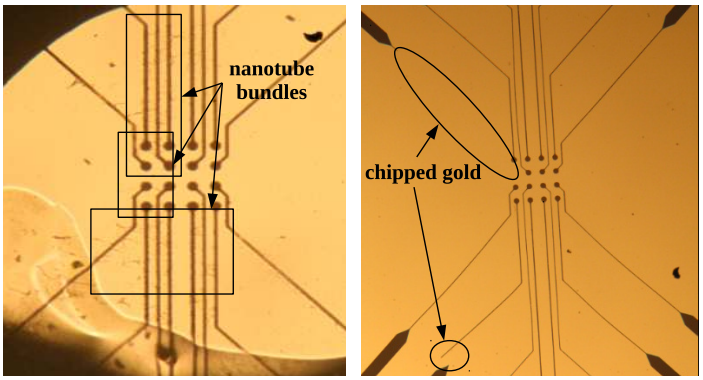
FIGURE 2.2: Observable defects on an electrode array prior to experimental use.

bottom) as they cover more surface. Only one wafer at a time can be placed in the thermal evaporator. Each array is used to test only one sample of material. Since they are not cleaned between tests, they are not affected by the cleaning process or possible left-over material as can be the case for the sixteen terminal micro-electrode array. The electrodes are therefore tested for defects once, at the end of the fabrication procedure.

Irrespective of the design, once arrays have been produced, they are placed under a microscope to identify any potential micro-scale defects that could affect the applied voltages or current measurements during experiments. Examples of defects observed on the sixteen terminal microelectrode array prior to use are illustrated in Figure 2.2. Some of the observable defects will clearly affect experiments (fig.2.2(c)), and the electrode is therefore marked as unusable (represented with a cross in the figure). However, other defects might have negligible impact on the experiments, despite being observable at micro-scale. In the case where no defects are observable at micro-scale, or where the defects can potentially have a negligible effect on the conductivity of the electrodes (fig.2.2(a) and (b)), a multimeter was used to test current flow across the contacts, and subsequently measure the electrode's resistance. It must be noted that only the larger part of the electrodes, furthest away from the material's contacts could be tested in this manner, leaving the possibility for unidentified defects prior to the start of experiments.

In the case of the sixteen-terminal arrays, the electrodes tend to degrade with time. This is due in part to chipping of the gold at the contacts with the edge connectors (when the array is removed and replaced a number of time), and in another part to the cleaning





(a) (b)

FIGURE 2.3: Observable nanotube left-overs and defects on two electrode arrays after they have been cleaned to remove material drop-cast in the previous experiment.

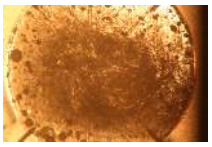
process. When dynamic materials are used, they are generally removed between experiments, and the slides are cleaned in an ultrasonic bath using propane-2-ol. Residues of composites are sometimes impossible to remove, whilst other times repeated cleaning removes both material and parts of the gold layer. In both cases, the quality of the electrode array is affected. Figure 2.3(a) and (b) present examples of bulk of SWCNTs left-over from the cleaning process and an array chipped due to cleaning, respectively.

As for the post-fabrication tests, optical tests are followed by electrical tests, in case defects are not visible at micro-scale. Prior to drop-casting the material on a cleaned slide, a multimeter is therefore used to test whether current flows across electrodes that should not be in contact, and the resistance across each electrode is subsequently measured. The lifespan of a sixteen terminal microelectrode array, in the sense that enough electrodes are still intact for an experiment to be undertaken, was found to be approximately twenty uses.

2.3 Apparatus

2.3.1 Electrical Characterisation

The I-V characteristics of the materials used in the EiM experiments were measured using a Keithley 2400 digital source meter. During the characterisation process, samples were kept in a screen metal chamber. This chamber was connected to a mechanical rotary vane vacuum pump. The pump was turned on for the solid samples, keeping them in constant 10^{-1} mbar conditions, and turned off for samples in liquid state, leaving them



in air. Output currents were measured across samples in steps. The applied voltage, or bias, was changed by 0.1 V each step, with a 1 s interval in-between. Since the board used in EiM experiments allows voltage levels between 0 V and 8.048 V, most characterisation tests had a lower and upper voltage limit of 0 V and 10 V, respectively.

2.3.2 Preparation

Solution storage

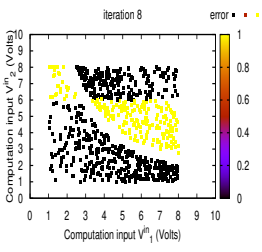
SWCNT-based composites were prepared and stored in glass vials with a capacity of $\approx$ 7.63 ml. Each vial was filled up to two third of its capacity during preparation and $\approx$ 20 μl of composite was used during each experiments. It was therefore possible to have samples extracted from the same original solution for about three months - given four experiments per day, five days a week, for one material, which is an over-estimate.

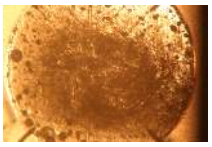
Since SWCNTs dispersed in a liquid matrix tend to aggregate over time [27], new composites needed to be produced frequently (approx. every 2 months) in order to ensure that all samples used in experiments presented similar electrical characteristics and level of dispersion. This justified the small dimensions of the vials, which helped limiting the amount of composite being wasted. It must be noted, however, that the process of renewing samples could result in differences in SWCNT concentrations, but these were too small to produce more than a negligible effect on the sample's electrical behaviour.

In the case of the other two materials, the memristors and the microtubules, it was observed that they tended to keep their electrical characteristics for a longer length of time if kept under vacuum. They were therefore stored in a screen metal chamber at constant 10^{-1} mbar conditions.

Electronic balance

An Ohaus Explorer Pro analytical balance was used to weigh the different components involved in the preparation of the majority of composites. This scale has a precision of 0.1 mg and it was placed in a negative pressure glove box. Considering the size of the vial used and the amount of material required, the high sensitivity of the scale was not sufficient to allow very low concentration of SWCNT-based composites to be prepared directly. Instead, 'mother solutions' of high SWCNT concentration were prepared first,



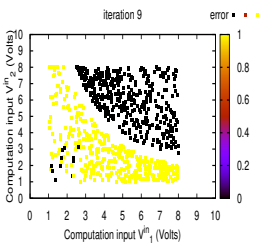


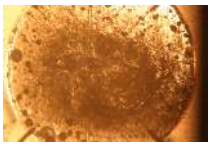
and subsequently diluted when low concentrations were required. A typical weight of 50mg of SWCNTs was used in the mother solutions.

Stirrer

A Cole Parmer 750 (W) ultrasonic processor with threaded probe and tapered microtip was used to stir the SWCNT-based composites. Tip sonication was chosen as it is faster than ultrasonic bath, and better for initial SWCNT dispersion than magnetic stirring [27], since the latter does not provide enough power to break the strong Van der Walls forces at the origin of SWCNT aggregation [28]. It must be noted, however, that the ultrasonic processor has the disadvantage of potentially damaging the SWCNTs if the intensity is too high [29]. In order to limit the damage to the SWCNTs, the processor was set at 20% intensity, with 5 s interval between stirrings. The potential introduction of impurities due to an unclean microtip, another disadvantage of tip sonication, was not seen as a major concern in the experiments undertaken. The SWCNTs used were not purified, and therefore impurities potentially modifying the behaviour of the composite were already present in the samples. In addition, the experimental set-up did not require the material to have electrical properties specific to purified SWCNT-based composites, since the identification of optimum properties for EiM materials remains under investigation.

The quality of the SWCNT dispersion was assessed visually. It was observed to vary depending on the concentration of SWCNTs and the choice of the matrix used. A longer sonication time was found necessary to disperse composites with higher SWCNT concentrations. This is consistent with the results reported in [27], although the exact formulation of the SWCNT-based composites differ. Once the initial dispersion was finished, a magnetic bead was added to the composites, and the glass vials were stored on a magnetic stirrer. This prevented early aggregation of the SWCNTs due to high concentrations or potentially poor initial dispersion, as suggested in [27]. Whilst some bundling of the SWCNTs is useful for the EiM experiments, as discussed in [30], the overall level of SWCNT dispersion should ideally be the same or similar in all samples investigated. The exact length of time the SWCNTs remained near-uniformly distributed in the composites was not investigated here. However, following this dispersion and storage method, large SWCNT aggregates appeared in micrographs after 1-4 months.





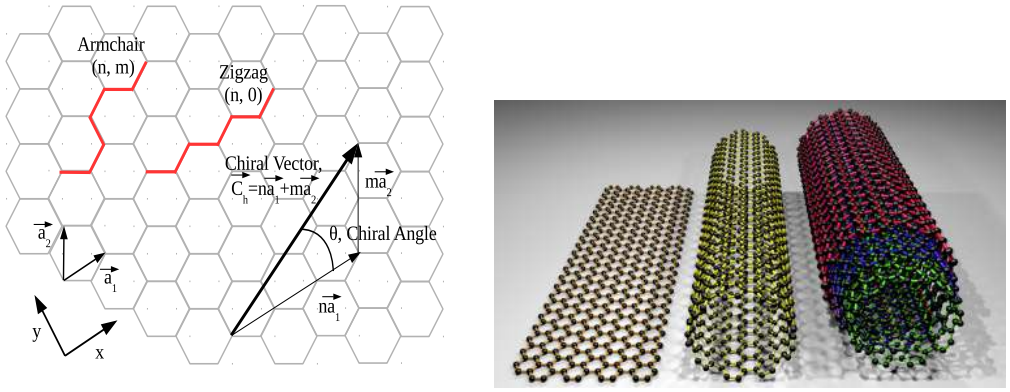
UV exposure

An Eprom Eraser UV141 was used to solidify SWCNT/epoxy composites. This eraser has a UV wavelength of 256.7 nm and light intensity of 5 mW/cm^2. The time taken to solidify the different SWCNT/epoxy samples using the Eprom Eraser varies according to the type of epoxy used and the SWCNT concentration. This is discussed later in the chapter.

2.4 Single-Walled-Carbon-Nanotube Composites

2.4.1 Carbon-Nanotubes

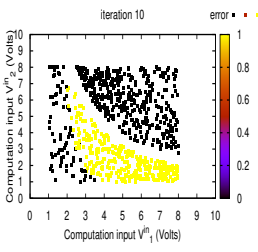
Carbon nanotubes (CNTs) are made of sp^2-hybridised carbon atoms arranged in hollow cylindrical structures [31]. CNTs can be single-walled (SWCNTs) or multi-walled (MWCNTs), according to the number of concentric cylinders they comprise. CNTs are closely related to graphene, as they are formed of the same hexagonal lattice of sp^2-hybridised carbon atoms. In the case of graphene, however, the lattice is flat rather than tubular. Figure 2.4 presents examples of a graphene sheet, a SWCNT and a MWCNT.

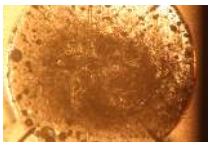


(a) (b)

FIGURE 2.4: (a) The different ways of folding a graphene sheet to produce SWCNT with different electrical characteristics (zigzag, armchair, chiral) and (b) from left to right: graphene sheet, single-walled and multi-walled CNTs

CNTs are considered 1-dimensional due to their high aspect ratio; lengths are of the order of a few micrometers whilst their diameter is of the order of tens of nanometers. The scale and electrical properties of CNTs vary according to the number of walls, their production method and the way they are folded [32]. MWCNTs are generally metallic, whilst SWCNT can be either metallic or semi-conducting, depending on the value of the chiral vector ($\vec{C_h}$ in Fig. 2.4(a)) along which they have been folded.



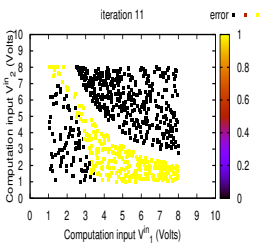


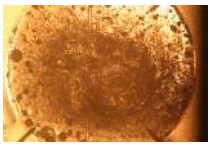
SWCNTs have unique thermal, mechanical and electrical properties that make them attractive in a number of fields, from structural engineering [33] to microelectronics [34, 35]. They have a charge carrier mobility up to $79,000$ cm^2/Vs, high current density capacity above 10^{13} A/m^2 and conductivity up to 5.1×10^5 S/cm [32]. These electronic properties make them an attractive candidate for EiM, specifically when mixed with polymers, epoxies or LCs: SWCNT-based composites can present complex non-linear input/output relationships, thereby providing a rich search space which can be explored and exploited using search algorithms.

SWCNTs in dry powder form were used in EiM experiments. They were purchased from Carbon Nanotechnologies Inc. (Houston,TX, USA). The powder contains carbon nanotubes which are 1/3 metallic and 2/3 semiconducting, with approximately 15% impurities. These are the specifications reported by the manufacturer and are typical of commercially available SWCNTs. No sorting or doping was undertaken before mixing the SWCNTs with other components to form the different composites. The differences in conductivity and the presence of impurities is likely to have an impact on the composites' electrical and physical characteristics [27]. However this impact on the results obtained during EiM experiments has not been investigated and all composites were prepared using the same SWCNTs, which is consistent with EiM literature [21, 36–38]. All SWCNT composites were prepared in a glove box due to their high aspect ratio.

2.4.2 Single-Walled-Carbon-NanoTubes in Polymer Matrix

The first nanotube-based composite investigated in EiM was a mixture of SWCNTs and poly (methylmethacrylate) (SWCNT/PMMA) [25, 39]. However, PMMA with a glass transition temperature at 105 $^\circ C$ is solid at room temperature. The electrical and computational behaviour of a SWCNT/PMMA composite is therefore only affected by changes in its bulk conductance. A different polymer, poly (buthylmethacrylate) (PBMA), was chosen subsequently as it presents a glass transition temperature at 20-25 $^\circ C$, offering the potential for SWCNTs to move within the bulk matrix under an applied electric field. In addition, SWCNT/PMMA dispersions were found to be less stable than those of SWCNT/PBMA. It is suggested in [22] that this difference relates to the length of the polymer chains. The longer chains of the PBMA make it more hydrophobic, and therefore easier to mix with the SWCNTs, themselves highly hydrophobic. Figure 2.5 shows





the chemical structure of the repeat unit of these two organic molecules, together with the structure of an anisole molecule, used as the solvent in the composite's preparation.

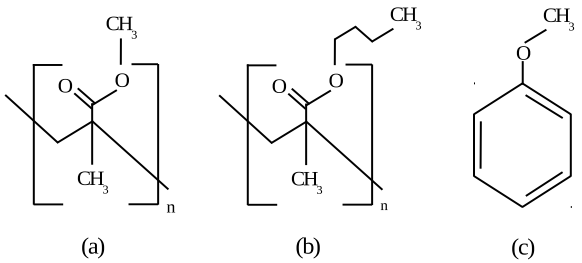


FIGURE 2.5: (a) poly(metyl metacrylate) (PMMA) ($[C_5H_8O_2]_n$), (b) poly(butyl metacrylate) (PBMA) ($[C_8H_{14}O_2]_n$) and (c) anisole (methoxybenzene, C_7H_8O)

Preparation

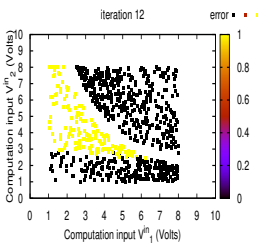
The method used to produce SWCNT/PBMA composites follows that originally developed for the SWCNT/PMMA mixtures. The PBMA was purchased from Merck, Japan. Preparation began with adding SWCNT in powder form to a glass vial placed on the analytical balance. The polymer in dry crystal form was subsequently added to the SWCNTs. Finally, the solvent (anisole) was drop-cast into the vial. When crystals were visibly dissolved, the solution was sonicated for $5 - 10$ min using the ultra-sonic probe, depending on the composite concentration, as discussed previously.

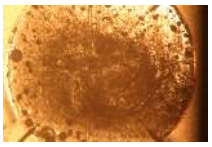
After the solution cooled down, a sample was extracted with a micro-pipette and drop-cast on an electrode array. The sample was left to dry, leaving a solid SWCNT / polymer layer on an electrode array. It must be noted that the solution in its liquid form was a mixture of SWCNT, polymer and anisole, with the weight $\%$ of SWCNT relating to the added weight of these three components. When a sample was deposited and the anisole evaporated, the weight $\%$ (wt $\%$) became:

$$\text{wt } \% = \frac{SWCNT(g)}{SWCNT(g) + PBMA(g)} \times 100(\%) \tag{2.1}$$

Characteristics of SWCNT/PBMA composites

The properties of SWCNT/PBMA composites have been investigated and reported in [22]. PBMA by itself possesses a very low electrical conductivity, as can be observed in Figure 2.6(a) where the current / voltage (I/V) curves of non-coated and PBMA covered eletrodes are compared. Their respective responses to the voltage sweep, from 0 to 5



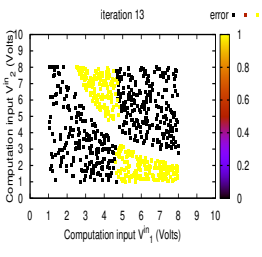


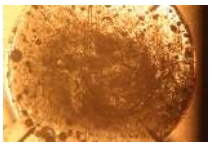
V and back to 0V, are almost undistinguishable. The negative current levels observed in the graph are due to an offset in the source meter. However, the level of the currents without this offset remains below 1.5×10^{-4} μA, which is too low to be picked up by the equipment used in the EiM experiments.

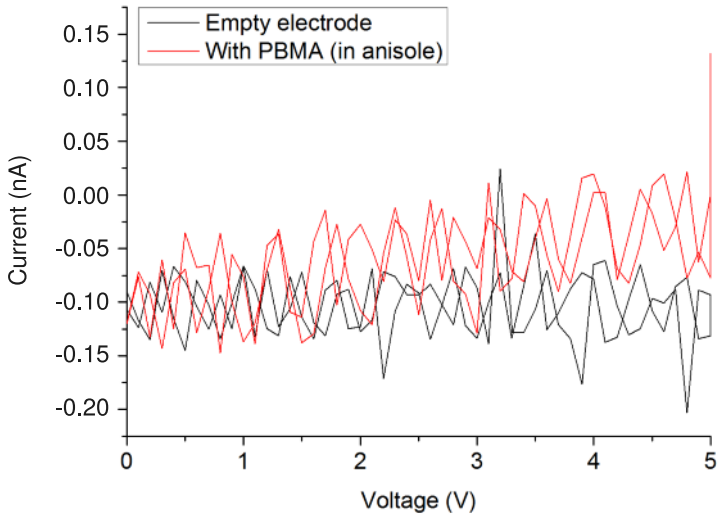
The role of the PBMA in SWCNT/polymer thin films is to introduce insulating areas between the nanotubes, as well as provide a matrix to define the morphology and keep the CNTs in place. This induces complex conduction mechanisms in the composite [22] and a non-linear I/V relationship depending on the SWCNT concentration. Figure 2.6 (b) and (c) show that composites with low SWCNT concentration present non-linear relationship between voltage and current, whilst past a 1 wt % SWCNT/PBMA threshold, the relationship becomes linear. It can also be seen in both sub-figures that the current increases with the SWCNT concentration.

The computational capabilities of SWCNT/PBMA samples have been tested for a variety of problems such as Boolean logic and even-parity [22, 41], with reproducible results. Composites with a concentration of $\approx$ 1 wt % SWCNT/PBMA were easiest to evolve, i.e. less time required to achieve the same or better computational results. This was found to be consistent with the composite's percolation threshold [23]. The latter corresponds to the minimum SWCNT concentration required for at least one conductive network to form, allowing a flow of charge carriers between electrodes.

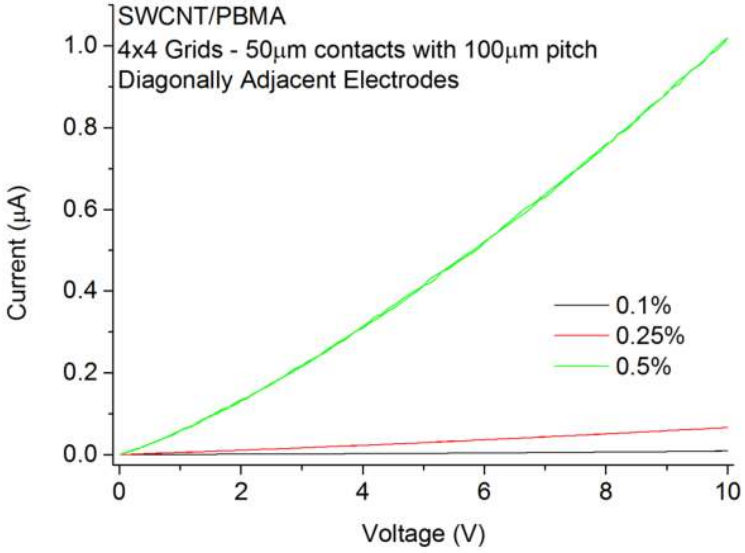
Above the composite's percolation threshold, the conductivity of the mixture sees a rapid increase. It has been observed that computational results obtained around this threshold are optimum, irrespective of the problem or framework implemented [22, 42, 43]. However, it was observed in [26] that it can take more time to evolve samples with higher concentration as compared to those with the critical 1 wt % SWCNT/PBMA concentration. It must be noted that whilst PBMA has been chosen due to its low glass transition temperature compared to PMMA, no visible movement of SWCNTs during the evolution process has been reported. SWCNT/PBMA composites are therefore considered solid (static) in this work.



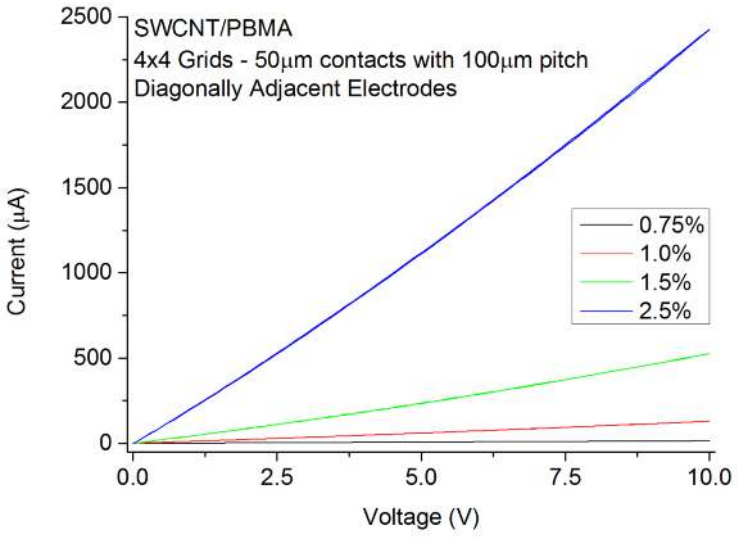




(a)

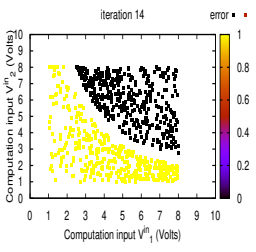


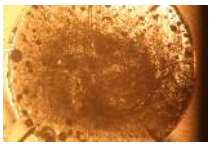
(b)



(c)

FIGURE 2.6: I/V characteristics of (a) non-coated (empty) electrodes and PBMA-only, (b) low and (c) high SWCNT/PBMA concentrations respectively ([40]).





2.4.3 Single-Walled-Carbon-NanoTubes in Liquid Crystals

LCs exist in a transition state between solid (anisotropic and ordered) and liquid (isotropic and disordered) called the mesomorphic state (mesophase). LCs generally present a degree of orientational order, but little translational order, as illustrated in Figure 2.7. They are first differentiated according to the way they reached the mesophase, either due to a temperature change or a change in concentration, resulting in thermotropic LCs or lyotropic LCs, respectively. Both types are divided in sub-categories characterised by the degree of order in molecule orientation, translation, and general geometry along which the singular molecules assemble [32]. Only lyotropic, rod-shaped nematic liquid-crystals are illustrated in Figure 2.7 whilst the others are reported by name.

LCs were chosen in the first EiM experiments due to their potential for transforming input signals into measurable output signals, but also for being reconfigurable and for working at a molecular level. These experiments were performed using a LC display (LCD) to evolve a tone discriminator [19]. Further investigations suggested that it was possible to perform other tasks such as Boolean logic and robot control [18, 20]. However, investigations of solutions evolved in the LCD highlighted that this material was unable to produce results that were consistent across tests and stable over long periods of time [9]. It was also unclear how much the LCD's electronics influenced the results.

Instead of LC arrays, the research presented here focused on an E7 nematic LC

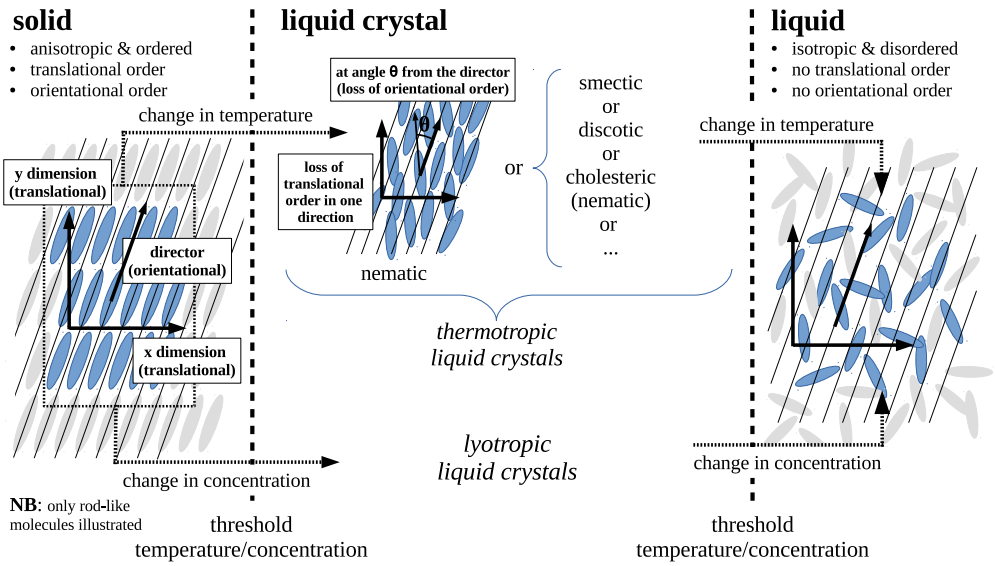
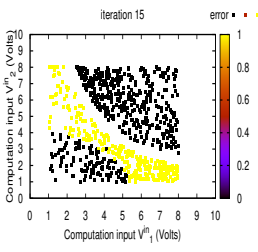
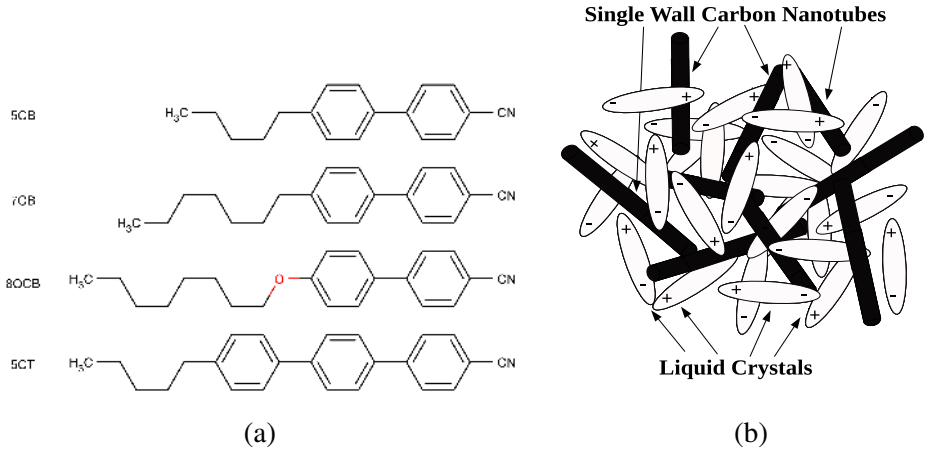


FIGURE 2.7: Transition of materials from solid to liquid crystalline to liquid, as affected by changes in temperature or concentration.



within which SWCNTS are dispersed. This results in a electrically conductive and physically dynamic blend which can be drop-cast on a micro-electrode array in the same way as SWCNT/polymer composites. Figure 2.8(a) illustrates the different molecules contained in this LC blend, whilst Figure 2.8(b) is a simplistic illustration of the structure of the SWCNT/LC composite before it has been subjected to an electric field.



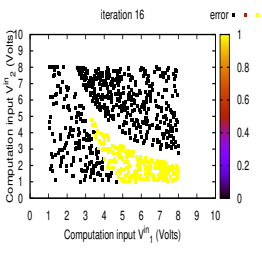
(a) (b)

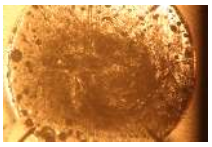
FIGURE 2.8: (a) Chemical structure of the E7 nematic liquid crystal (LC) molecules purchased fom Merk Japan and (b) simple schematic representation of the SWCNT/LC blend.

Preparation

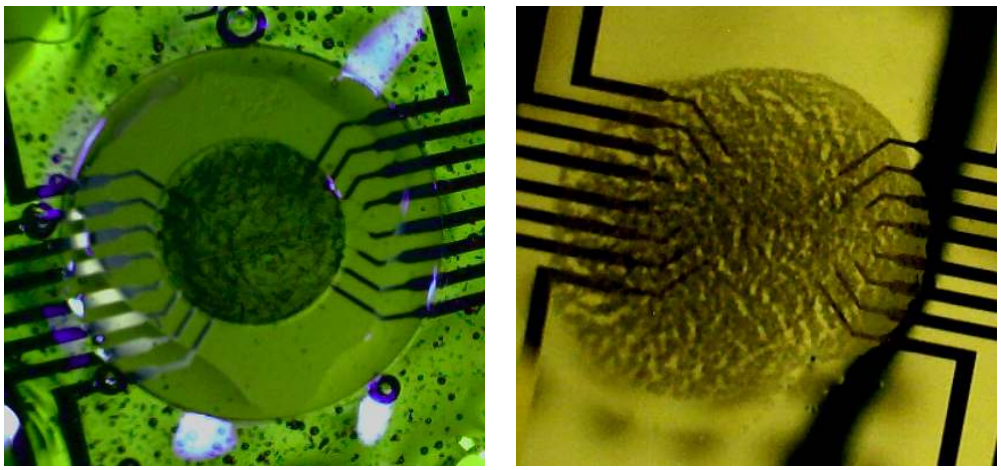
The concentration of nanotubes in SWCNT/LC composites was quantified using weight %. Preparation began with adding SWCNTs powder to a glass vial placed on an analytical balance. The LC was drop-cast on the nanotubes using a micro-pipette up to a given weight. The solution was subsequently sonicated for 30-60 seconds until the dispersion appeared homogeneous to the naked eye.

About 20 μl of SWCNT/LC composite was extracted from the glass vial with a micropipette and drop-cast within a nylon washer of 2.5 mm internal diameter. The washer was previously fixed on the electrode array using a two part epoxy resin. The disadvantage of using a washer is that the thickness of the SWCNT/LC film is thicker than when no washer is used, and consequently, it was more difficult to record precise images of the material using simple light microscopy imaging. However, using a washer to contain the liquid samples during experiments, allowed changes in the morphology to be linked to variations in the electric field to which the samples were subjected, rather than variation in the sample's bulk geometry.





Examples of photographs taken with and without a washer are presented in Figure 2.9(a) and (b) respectively. In both cases the photographs were taken within $60s$ of the sample being drop-cast, and before any electric field had been applied. It can be noticed from Figure 2.9(b) that when the sample is not contained within a washer, a layer of low SWCNT concentration forms on the edge of the sample.



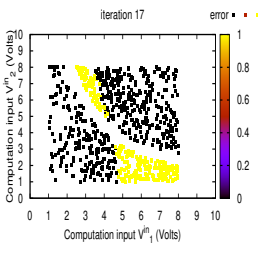
(a) (b)

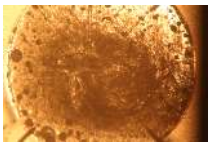
FIGURE 2.9: Microscope photographs of 0.05 wt % SWCNT/LC drop-cast on a micro-electrode array (a) within a 2.5 mm nylon washer and (b) directly on the array.

Electrical characteristics of SWCNT/LC composites

Testing the I/V characteristics of LC-only samples drop-cast on the two terminal electrode arrays showed that the E7 blend had a conductivity similar to that of bare electrodes [30]. In subsequent experiments, the I/V characteristics of different SWCNT/LC concentrations were measured using the same equipment and electrode design. The current outputs collected across a 0.05 wt % SWCNT/LC sample subjected to ten consecutive voltage sweeps from 0V to 10V are plotted in Figure 2.10 (a), whilst Figure 2.10 (b) presents the results of five consecutive sweeps on a 0.5 wt % SWCNT/LC sample.

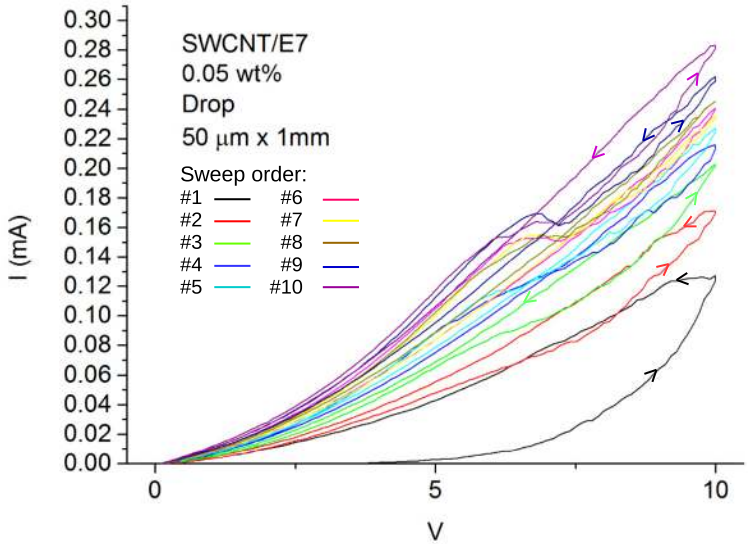
Comparing the two graphs, it can be observed that the maximum output current is higher with the higher SWCNT concentration. On the other hand, the I/V relationship is more non-linear for the lower SWCNT concentration. In both cases, an anti-clockwise hysteresis can be observed, with currents being lower when the voltages are increased to $10V$ than when they go back to $0V$, as illustrated by arrows in Figure 2.10. This suggests the presence of a charge trapping mechanism in the samples, and more specifically, the trapping of negatively charged particles, i.e. electrons. Whilst this mechanism has not



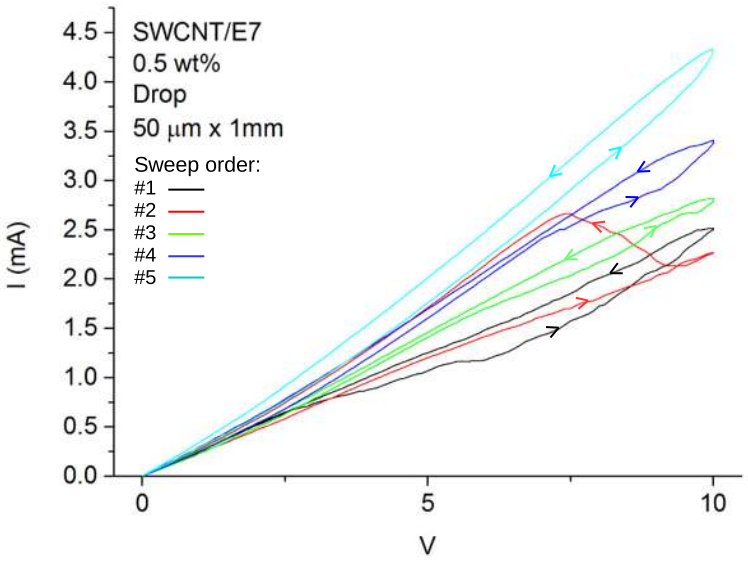


been further investigated here, it shows a difference in terms of electrical behaviour between SWCNT/LC composites and SWCNT/PBMA composites (Fig.2.6), and can therefore be ascribed to the liquid matrix. In addition, under the influence of the same voltage level, SWCNT/LC samples produced higher currents than solid SWCNT/PBMA samples for lower SWCNT concentrations.

This suggests that the percolation threshold is lower for ordered CNTs in the liquid material, which can be explained by the fact that SWCNTs in LCs tend to bundle under an applied electric field, establishing percolation paths between electrodes [30]. Applying an electric field to a SWCNT/LC sample can modify the arrangement of SWCNTs

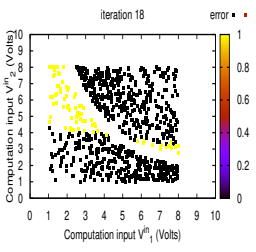


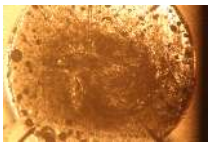
(a)



(b)

Figure 2.10: I/V characteristics of (a) 0.05 wt % and (b) 0.5 wt % SWCNT/E7 nematic LC samples ([40]).





in the matrix allowing them to form conductive pathways between electrodes that might not have been present when the sample was drop-cast on the electrode array. This is not possible in SWCNT/PBMA composites, where the physical connections between nanotubes cannot be modified once the material has solidified. The electrical characteristics of the latter material are therefore dependant on its physical state post anisole evaporation, and the solution space remains unchanged throughout training.

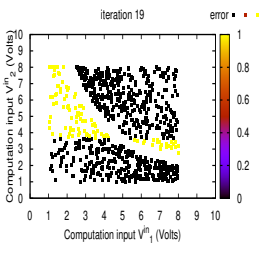
Whilst LC molecules are able to orient single nanotubes, it is suggested in [30] that this is no longer possible when a SWCNT/LC sample is subjected to an electric field, due to the greater length of the SWCNT bundles created as compared to the LC molecules. However, the LC matrix enables the SWCNTs to form complex networks which can vary depending on the applied electric field. This adds an extra dimension to the EiM problem compared to the case where SWCNT-based composites are in a solid state [21, 22]. The purpose of the LC matrix is therefore to: 1) create insulating areas in the composite, in the same way as PBMA, and 2) provide a matrix within which the SWCNTs for complex and reconfigurable networks under an applied electric field.

In summary, when SWCNT/LC composites are used, not only the material's electrical properties but also its morphology can be changed using specific signals [44]. Preliminary results characterising the computational capabilities of SWCNT/LC samples to solve computation problems using EiM were reported in [37].

2.4.4 Single-Walled-Carbon-Nanotubes in Epoxy Matrix

Samples of SWCNT/epoxy composites have been investigated here for the first time in the context of EiM, owing to their capacity to solidify, combining advantages of liquid and solid SWCNT-based mixtures. The main components of epoxy resins are epoxide groups (at least two in any formulation) and a hardener or curing agent (which is often an amine molecule) [45, 46]. An example of the structure formed when two of these molecules react together is illustrated in Figure 2.11.

In two-part epoxies, a specific ratio of the two molecules must be mixed together for them to harden. This curing process is not ideal for training as, once mixed, the two-part epoxy solidifies, leaving only a specific time span during which experiments can take advantage of the liquid nature of the samples. In addition, it is difficult to determine



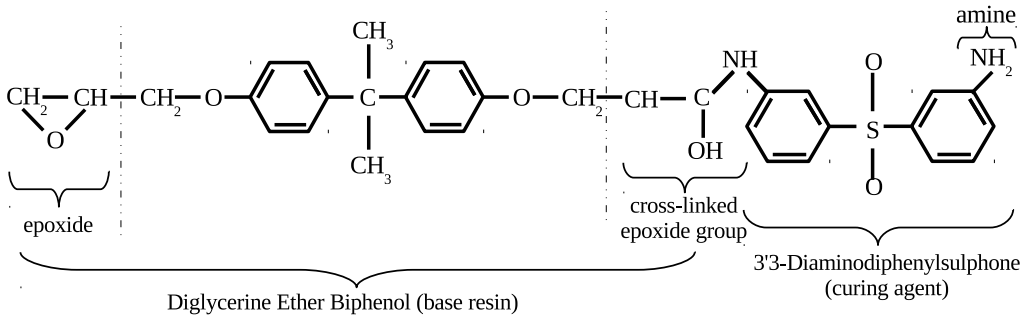


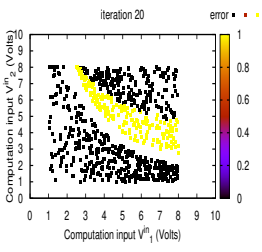
FIGURE 2.11: Chemical structure of epoxy components [47]

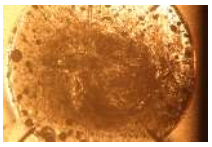
the influence of the curing process on the final solution when curing and algorithm-controlled evolution occur simultaneously.

In one part epoxies, the epoxide group and the hardener are already mixed. The binding of the two molecules is initiated by an external stimulus. Such a stimulus can be UV light. One-part epoxies with thermosetting properties for example can be cured via irradiation. Once exposed to UV light, the epoxy group opens and allows cross-linking with the hardener, changing the mixture irreversibly into infusible, insoluble polymers.

Results reported in [48] show that it is possible to align carbon nanotubes dispersed in a one-part epoxy matrix using an electric field. Moreover, this manipulation results in modified electrical properties of the composite. This suggests that it should be possible to evolve liquid SWCNT/epoxy samples using the same experimental set-up as for SWCNT/LC composites and subsequently cure them, resulting in computing devices that can be physically manipulated.

Experiments reported here were undertaken using two different one-part UV cure epoxies as component of nanotube based composites. First attempts were based on a katiobond LP655 UV-cure epoxy purchased from Delo Adhesives. It has a viscosity of 12000 cps at room temperature ($\approx 23^{\circ}$C) and recommended irradiation time of 20s, using 400 nm LED with an of intensity 200 mW cm^{-2}, with a curing time of 24 h post irradiation to reach final strength [49]. The choice of the second epoxy, NO81, was directed by the results obtained with the first, and specifically the need for a less viscous and faster cure material. The NO81 epoxy was purchased from Norland products incorporated. It has a viscosity of 300 cps at 25°C and recommended irradiation time is on average 15 s using a 365 nm LED with an intensity of 2 W cm^{-2} [50, 51].



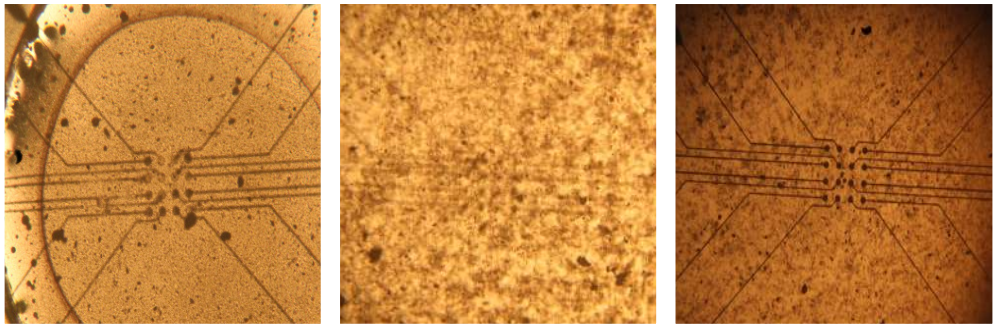


Preparation

SWCNTs were less easily dispersed in the epoxies than in the LC or PBMA/anisole solutions. To aid with dispersion, half of the epoxy used in the preparation was first drop-cast into a glass vial. The total weight of SWCNTs in dry powder form was then added to the vial before drop-casting the second half of the epoxy.

When the more viscous LP655 epoxy was used, sonication was performed in two sets of 10 min, with a 5 min interval where the mixture was left to cool down. Using the less viscous NO81 epoxy, sonication was performed in two sets of 3 minutes.

In both cases, dispersions were drop-cast within washers fixed on the electrodes and either left liquid, or cured using the Eprom Eraser with a power of 5 mWcm^{-2} and 253.7 nm UV wavelength. It must be noted that the curing time was observed to depend on the SWCNT concentration. As the SWCNT concentration increased, so did the UV exposure time required for the composite to solidify. This will be discussed in Chapter 6. The micrographs of 0.05 wt% SWCNT/epoxy samples are presented in Figure 2.12, where (a) and (b) are in liquid form and (c) is taken after (b) has been cured.

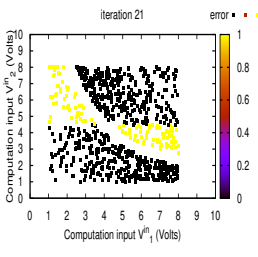


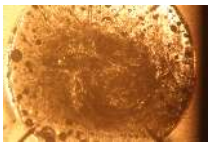
(a) (b) (c)

Figure 2.12: Liquid samples of 0.05 wt % (a) SWCNT/LP655 and (b) SWCNT/NO81 drop-cast on two micro-electrode arrays and (c) cured version of the SWCNT/NO81 sample.

Electrical characteristics of SWCNT/epoxy composites

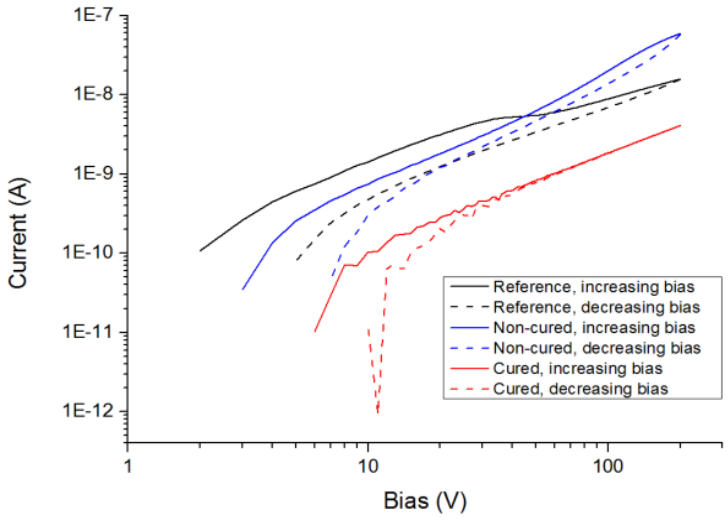
The set of results presented in Figure 2.13(a) illustrate the I/V characteristics of a LP655 sample dropcast on a two terminal electrode array and subjected to voltage sweeps from 0 to 200 V and back. A set of empty electrodes was used as a reference, and is labelled as such on the graph. It presents a non-zero, but negligible, conduction, a behaviour previously observed in Fig.2.6(a) and discussed Sec.2.4.2. Within the voltage levels of interest for experiments, i.e. within 0 V and 10 V, it can be observed that the LP655



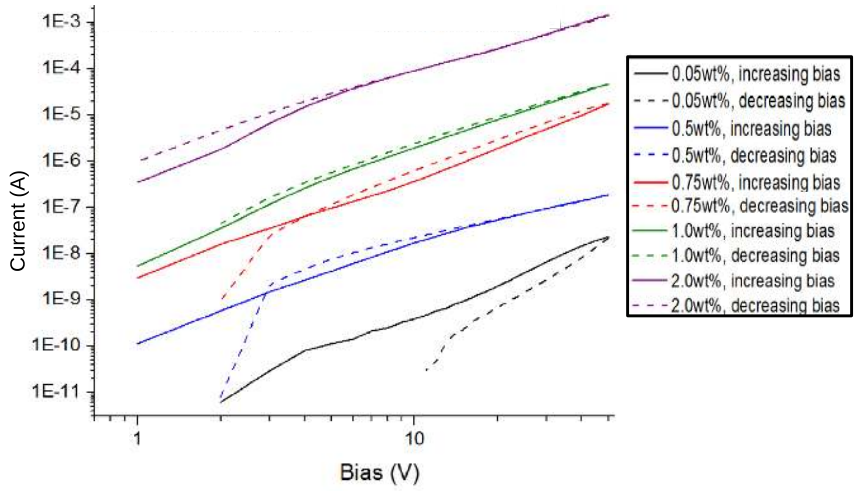


sample in both liquid (non-cured) and solid (cured) state presented current outputs lower that 1 $\times 10^{-8}$A. This is below the sensitivity threshold of the EiM hardware motherboard and similar to the output levels of the reference, as observed in Figure 2.13(a). The sample's conductivity for these voltages can therefore be considered negligible. It must be noted that the current outputs measured across the LP655 sample in liquid state were always higher than post-curing. This suggests that charges tend to move more easily across the LP655 prior to curing.

When SWCNTs were added to LP655 samples in liquid state, their conductivity tended to increase with increasing SWCNT concentration. However, in Figure 2.13(b), the liquid composites with a concentration of 0.05 wt % SWCNT/LP655 presented the

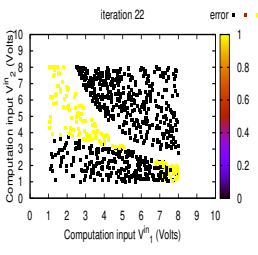


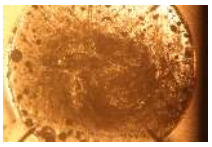
(a)



(b)

Figure 2.13: I/V characteristics of (a) LP655 pre and post-curing under UV light and (b) multiple concentrations of SWCNT/LP655 in liquid state (pre-curing) ([52]).



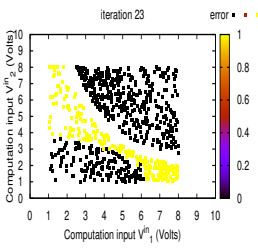


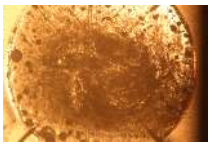
same I/V characteristics as the liquid LP655 sample from Fig.2.13(a). A minimum amount of added SWCNTs appears to be required for the composite to have an electrical behaviour different from pure liquid LP655, at least prior to EiM training.

Over all concentrations, results presented in Figure 2.13(b) suggest that, for the same applied voltage, SWCNT/LP655 composites were not as conductive as either solid SWCNT/PBMA (Fig. 2.6) and liquid SWCNT/LC (Fig. 2.10) composites. For example, the current output of a 0.5 wt % SWCNT/LP655 sample under a 10V bias was 1.5×10^{-8}A (15nA), whilst 1.1×10^{-6}A (1.1μA) and 2.55×10^{-3}A (2.55mA) were recorded across the 0.5 wt % SWCNT/PBMA (Fig. 2.6(c)) and 0.5 wt % SWCNT/LC samples (Fig. 2.10(b)) under the same applied voltage, respectively. However, past 0.5 wt % SWCNT/LP655, it was possible to achieve current outputs above the motherboard's sensitivity threshold when the bias was between 0 and 10V, thus satisfying the necessary requirement for the use of the SWCNT/LP655 in EiM experiments.

Beyond satisfying the measurability requirement, the electrical behaviours reported in Fig.2.13(b), added to the liquid state of the material, confirmed the potential for reproducing results obtained with the SWCNT/LC composites using liquid SWCNT/LP655. As for samples of SWCNT/LC material, the I/V curves obtained with the non-cured SWCNT/LP655 samples were non-linear for all concentrations, and an anti-clockwise hysteresis could be observed above 0.05 wt %. The same behaviours were observed after the samples had been cured under UV-light, but output current levels were reduced by up to one order of magnitude for each concentration. This is similar to the observations made with pure LP655, and suggests that despite the presence of the SWCNTs, the ability of percolation paths to form across the electrodes changes after cross-linking of the epoxy group with the hardener, potentially affecting solutions evolved during EiM.

The NO81 epoxy was chosen as a replacement for the LP655 following observations made during EiM experiments undertaken with SWCNT/ LP655 composites. The electrical characteristics of the liquid and solid SWCNT/NO81 were not measured on two terminal devices in the same way as the other materials, and are therefore not presented here. However, currents were measured across the samples used during the course of the EiM experiments. These measurements showed that the NO81 composites presented higher current outputs compared to the SWCNT/LP655 for the same concentrations. In fact, 0.05 wt% liquid SWCNT/NO81 composites were able to obtain currents similar to





those obtained with 0.05 wt% SWCNT/LC composites under the same voltage levels. This suggests that percolation paths were easier to form in the lower viscosity matrix of the NO81 epoxy . In addition, compared to the SWCNT/LP655 samples, curing the SWCNT/NO81 samples did not result in a reduction of the current output levels to one order of magnitude. This last observation is to be taken with care, however, as the measurements performed post-curing were taken across evolved SWCNT/NO81 composites, where SWCNT structures had been modified continuously through repeated application of algorithm-controlled voltages, which was not the case for the SWCNT/LP655 measurements reported here.

2.5 Linear Resistors Array

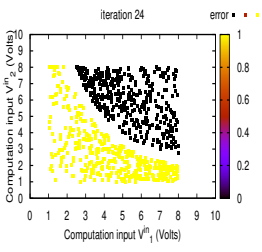
An array of resistors was designed as a reference device, with I/V characteristics aiming to match the range of current outputs produced by the SWCNT-based composites. The Ohmic resistors making up this array are sometimes referred to as linear resistors in this work, as they present the linear I/V relationship characteristic of such components. The rationale behind the fabrication and use of this array is to test whether or not the different materials investigated bring an advantage over a more conventional device.

Preparation

The resistor array was fabricated on a microscope slide using etch-back photolithography, following the same procedure as described in Section 2.2.2 when producing the different electrode arrays. A second array with the same characteristics was fabricated using 4.7 kΩ resistors soldered on a vero board. Figure 2.14 presents the circuit diagram for this array, as well as photographs of the mask used when exposing the positive photoresists during the photolithography process and the array fabricated on the vero board.

Electrical characteristics of linear resistor array

The array presents a linear I/V relationship. The electrode pairs (A, B and C in Figure 2.15) consisted of two electrodes of the device presented in Figure 2.14, and randomly selected out of all possible electrode combinations. The current outputs are dependent on



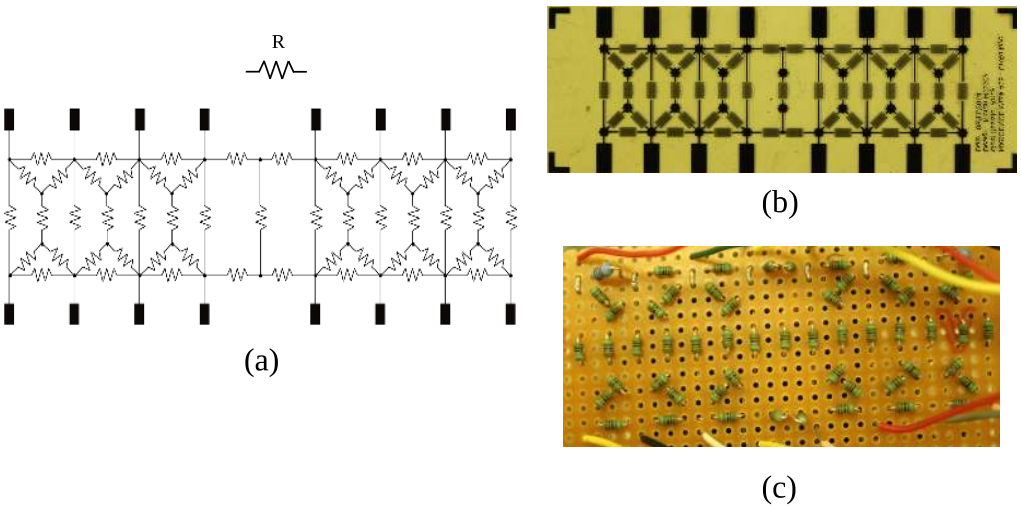


Figure 2.14: Resistor array (a) circuit diagram, (b) mask for etch-back photolithography and (c) fabricated on a vero board using resistors.

the pair across which they are measured, which also depends on the distance and number of resistors between the pair selected. The resistor array was originally fabricated to present similar output current levels to the SWCNT/polymer composites. However, when comparing Figure 2.15 to Figures 2.6 and 2.10, it can be seen that the range of currents produced by the array under a voltage sweep from 0 to 10 V matches better the current levels collected from the SWCNT/LC composites of 0.05 wt % to 0.5 wt %. The resistor array is thus well suited as a reference device for the unconventional materials proposed here, and especially for the SWCNT/LC samples.

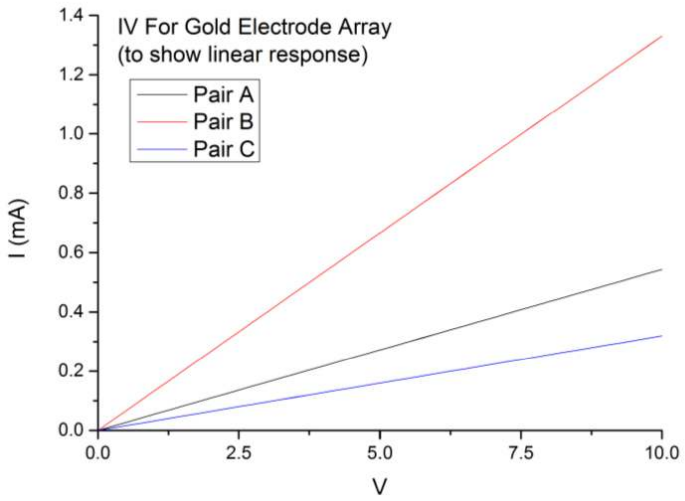
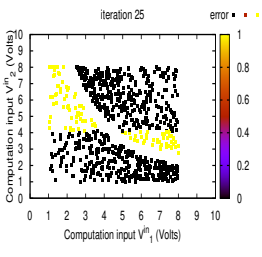
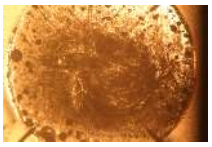


FIGURE 2.15: I/V characteristics of an array of linear resistors ([40]).





2.6 Summary of the Characteristics and Suitability of Chosen Materials

The materials described in this Chapter were selected according to three main factors: 1) they do not involve silicon, 2) they are the subject of numerous investigations within the fields of electronics and unconventional computing, yet work remains to be done before they can be integrated within the current technology or compete with it and 3) they present a potential for being evolved using the EiM framework.

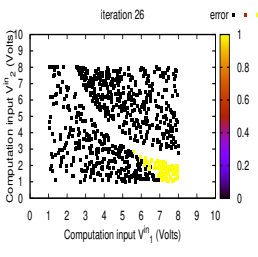
Carbon Nanotubes (CNTs) have been suggested as an alternative to silicon since their discovery [31]. The first CNT-based computer was reported in 2013 [53] and important advances have been made since [54, 55]. However, to the author's knowledge, fully integrated circuits based on CNTs and rivalling current silicon technology have not yet been reported. This motivates investigations into unconventional approaches which, as suggested in [17, 56] might provide a cheap and efficient framework to produce CNT-based devices.

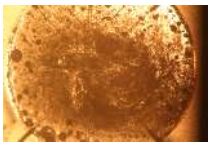
The array of linear resistors was designed to be used as a control device in order to assess the potential of the various materials, and specifically the SWCNT-based composites.

The main characteristics of the different materials used are presented in Table 2.1. The aim of this table is to provide a quick overview of material characteristics, such as the linearity (L) or non-linearity (NL) of the current (I)/ voltage (V) curve, identified as crucial for a material's use in the EiM framework. In conclusion, all materials, except the resistor array, present relatively complex electrical characteristics (non-linear and in some cases hysteretic I / V curves). The amorphous state of the liquid composites adds to this complexity as compared to solid samples. Overall, a complex behaviour allows a diversity of response under a given input that has the potential to be exploited by an

TABLE 2.1: Summary of the characteristics of materials for evolution in materio.

material	electrical characteristics	state	substrate origin
SWCNT/PBMA	NL I/V 0.5%<[]<1.5%	Solid	non-bio
SWCNT/LC	NL I/V, 0.05%<[]<1%	Liquid	non-bio
SWCNT/epoxy	NL I/V, 0.05%<[]<2%	liquid/solid	non-bio
resistor array	L I/V	Solid	non-bio





algorithm in the search for the solution to a computational problem. On the other hand, this complexity in behaviour makes it difficult to model the materials, and to use deterministic algorithms to find solutions to computational problems. The EiM framework allows materials to be treated as black boxes which properties can be exploited by non-deterministic algorithm, with only a knowledge of the boxes' inputs and outputs. The next chapter, Chapter 3, details the software and hardware implementation that allows computer-controlled evolution to transform materials into computational devices.

Bibliography

[1] L. Floridi, "Open problems in the philosophy of information," *Metaphilosophy*, vol. 35, no. 4, pp. 554–582, 2004.

[2] G. Piccinini and A. Scarantino, "Information processing, computation, and cognition," *Journal of biological physics*, vol. 37, no. 1, pp. 1–38, 2011.

[3] C. Horsman, S. Stepney, R. C. Wagner, and V. Kendon, "When does a physical system compute?," *Proceedings of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 470, no. 2169, 2014.

[4] NASCENCE project (ICT 317662), "Report on materials systems and electrical behaviour," 2013. Deliverable D1.1.

[5] J. F. Miller and K. Downing, "Evolution in materio: Looking beyond the silicon box," *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on*, pp. 167–176, 2002.

[6] A. Thompson and P. Layzell, "Analysis of unconventional evolved electronics," *Communications of the ACM*, vol. 42, no. 4, pp. 71–79, 1999.

[7] S. L. Harding and J. F. Miller, "Evolution in materio," *Encyclopedia of complexity and systems science*, 2009.

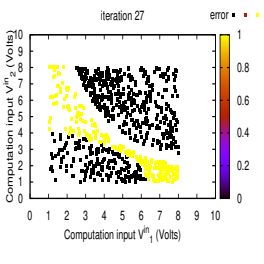
[8] J. F. Miller, S. L. Harding, and G. Tufte, "Evolution-in-materio: evolving computation in materials," *Evolutionary Intelligence*, vol. 7, no. 1, pp. 49–67, 2014.

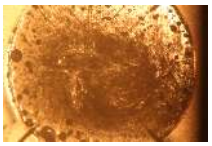
[9] S. L. Harding, J. Neil, K. Zauner, and J. Miller, "A Framework for the Automatic Identification and Extraction of Computation from Materials," *Computer*, 2008.

[10] S. Nichele, D. Laketic, and G. Tufte, "Is there chaos in blobs of carbon nanotubes used to perform computation?," *7th International Conference on Future Comp. Tech. and Applications, IN PRESS*, 2015.

[11] O. R. Lykkebø and G. Tufte, "Evolution-in-materio of a dynamical system with dynamical structures," in *Artificial Life Conference*, p. 242, 2016.

[12] D. Laketic, G. Tufte, O. R. Lykkebo, and S. Nichele, "An explanation of computation-collective electrodynamics in blobs of carbon nanotubes," in *Proceedings of the 9th EAI International Conference on Bio-inspired Information and Communications Technologies (formerly BIONETICS)*, pp. 43–48, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2016.





[13] D. Laketic and G. Tufte, "Evolution-in-materio computations: Hierarchies rising from electron dynamics in carbon nanotubes," in *Proceedings of the European Conference on Artificial Life 2017*, 2017.

[14] M. Amos, I. M. Axmann, N. Blüthgen, F. de la Cruz, A. Jaramillo, A. Rodriguez-Paton, and F. Simmel, "Bacterial computing with engineered populations," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 373, 2015.

[15] A. Adamatzky, "Thirty seven things to do with live slime mould," in *Advances in Unconventional Computing*, pp. 709–738, Springer, 2017.

[16] A. Tero, S. Takagi, T. Saigusa, K. Ito, D. P. Bebber, M. D. Fricker, K. Yumiki, R. Kobayashi, and T. Nakagaki, "Rules for biologically inspired adaptive network design," *Science*, vol. 327, no. 5964, pp. 439–442, 2010.

[17] S. Stepney, "The neglected pillar of material computation," *Physica D: Nonlinear Phenomena*, vol. 237, no. 9, pp. 1157–1164, 2008.

[18] S. L. Harding and J. F. Miller, "Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal," *Evolvable Systems: From Biology to Hardware*, pp. 155–164, 2005.

[19] S. L. Harding and J. F. Miller, "Evolution in materio: A tone discriminator in liquid crystal," *Congress on Evolutionary Computation, 2004. CEC2004.*, vol. 2, pp. 1800–1807, 2004.

[20] S. L. Harding and J. F. Miller, "Evolution in materio: Evolving logic gates in liquid crystal," *Proc. Eur. Conf. Artif. Life (ECAL 2005), Workshop on Unconventional Computing: From cellular automata to wetware*, pp. 133–149, 2005.

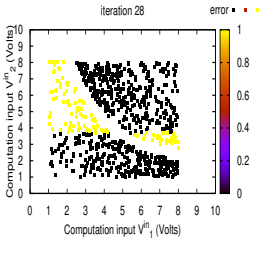
[21] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes.," *International Journal of Unconventional Computing*, vol. 10, no. 5-6, pp. 473–497, 2014.

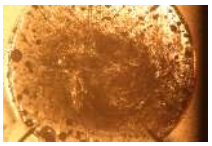
[22] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[23] F. Qaiser, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, and M. C. Petty, "Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation," in *IEEE Congress on Evolutionary Computation (CEC)*, pp. 5255–5261, IEEE, 2016.

[24] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebo, M. K. Massey, and M. C. Petty, "Evolution-in-materio: A frequency classifier using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 46–53, IEEE, 2014.

[25] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving function optimization problems using materials," in *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pp. 1–8, IEEE, 2014.





[26] NASCENCE project (ICT 317662), "Report on computational capabilities," 2014. Deliverable D1.2.

[27] S. Schymura, *Liquid Crystalline Carbon Nanotube Suspensions - From Unique Challenges to Unique Properties*. PhD thesis, 07 2013.

[28] S. Schymura, M. Kühnast, V. Lutz, S. Jagiella, U. Dettlaff-Weglikowska, S. Roth, F. Giesselmann, C. Tschierske, G. Scalia, and J. Lagerwall, "Towards efficient dispersion of carbon nanotubes in thermotropic liquid crystals," *Advanced Functional Materials*, vol. 20, no. 19, pp. 3350–3357, 2010.

[29] S. Badaire, P. Poulin, M. Maugey, and C. Zakri, "In situ measurements of nanotube dimensions in suspensions by depolarized dynamic light scattering," *Langmuir*, vol. 20, no. 24, pp. 10367–10370, 2004.

[30] D. Volpati, M. K. Massey, D. Johnson, A. Kotsialos, F. Qaiser, C. Pearson, K. Coleman, G. Tiburzi, D. A. Zeze, and M. C. Petty, "Exploring the alignment of carbon nanotubes dispersed in a liquid crystal matrix using coplanar electrodes," *Journal of Applied Physics*, vol. 117, no. 12, p. 125303, 2015.

[31] S. Iijima, "Synthesis of carbon nanotubes," *Nature*, vol. 354, no. 6348, pp. 56–58, 1991.

[32] M. C. Petty, *Organic and Molecular Electronics: From Principles to Practice*. New York, NY, USA: Wiley, 2018.

[33] B. Chen, S. Li, H. Imai, L. Jia, J. Umeda, M. Takahashi, and K. Kondoh, "Load transfer strengthening in carbon nanotubes reinforced metal matrix composites via in-situ tensile tests," *Composites Science and Technology*, vol. 113, pp. 1 – 8, 2015.

[34] A. Sleiman, P. Sayers, D. a. Zeze, and M. Mabrook, "Two-terminal organic non-volatile memory (onvm) devices," in *Handbook of Flexible Organic Electronics: Materials, Manufacturing and Applications*, pp. 413–428, 12 2015.

[35] F. Fuchs, A. Zienert, C. Wagner, J. Schuster, and S. Schulz, "Interaction between carbon nanotubes and metals: Electronic properties, stability, and sensing," *Microelectronic Engineering*, vol. 137, pp. 124 – 129, 2015. Materials for Advanced Metallization 2014.

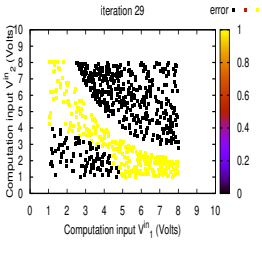
[36] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.

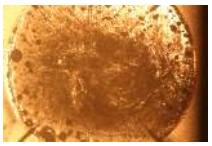
[37] M. K. Massey, A. Kotsialos, D. Volpati, E. Vissol-Gaudin, C. Pearson, L. Bowen, B. Obara, D. A. Zeze, C. Groves, and M. C. Petty, "Evolution of electronic circuits using carbon nanotube composites," *Scientific reports*, vol. 6, 2016.

[38] M. Dale, S. Stepney, J. F. Miller, and M. Trefzer, "Reservoir computing in materio: A computational framework for in materio computing," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2178–2185, IEEE, 2017.

[39] K. D. Clegg, J. F. Miller, K. Massey, and M. Petty, "Travelling salesman problem solved 'in materio'by evolved carbon nanotube device," in *International Conference on Parallel Problem Solving from Nature*, pp. 692–701, Springer, 2014.

[40] M. K. Massey, "Internal durham nascence meeting reports." Internal report, unpublished, 2013-2016.





[41] M. Mohid and J. Miller, "Solving even parity problems using carbon nanotubes," in *Computational Intelligence (UKCI), 15th UK Workshop on. IEEE Press*, 2015.

[42] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[43] M. Dale, *Reservoir Computing In-Materio (PhD Thesis)*. University of York, 2018.

[44] M. Massey, D. Volpati, F. Qaiser, A. Kotsialos, C. Pearson, D. Zeze, and M. Petty, "Alignment of liquid crystal/carbon nanotube dispersions for application in unconventional computing," in *AIP Conference Proceedings*, vol. 1648, p. 280009, AIP Publishing, 2015.

[45] B. Ellis *et al.*, *Chemistry and technology of epoxy resins*. Springer, 1993.

[46] N. Composites. https://netcomposites.com/guide-tools/guide/resin-systems/epoxy-resins/, 2018.

[47] T. Okabe, Y. Oya, K. Tanabe, G. Kikugawa, and K. Yoshioka, "Molecular dynamics simulation of crosslinked epoxy resins: curing and mechanical properties," *European Polymer Journal*, vol. 80, pp. 78–88, 2016.

[48] Y.-F. Zhu, C. Ma, W. Zhang, R.-P. Zhang, N. Koratkar, and J. Liang, "Alignment of multiwalled carbon nanotubes in bulk epoxy composites via electric field," *Journal of Applied Physics*, vol. 105, no. 5, p. 054319, 2009.

[49] D. Adhesives. https://www.delo-adhesives.com/fileadmin/datasheet/DELO%20KATIOBOND_LP655_%28TIDB-GB%29.pdf, 2018.

[50] N. P. Incorporated. https://www.norlandprod.com/adhesives/NOA%2081.html, 2016.

[51] N. P. Incorporated. https://www.norlandprod.com/msds/noa%2081msd.html, 2017.

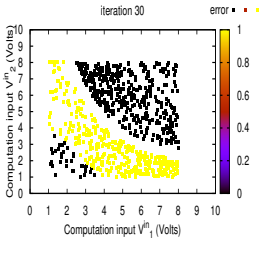
[52] M. Pillot, "Electrical properties of carbon nanotubes / epoxy composites." Internal report, unpublished, 2016.

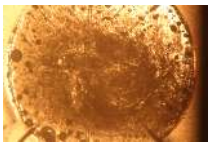
[53] M. M. Shulaker, G. Hills, N. Patil, H. Wei, H.-Y. Chen, H.-S. P. Wong, and S. Mitra, "Carbon nanotube computer," *Nature*, vol. 501, no. 7468, p. 526, 2013.

[54] Q. Cao, J. Tersoff, D. B. Farmer, Y. Zhu, and S.-J. Han, "Carbon nanotube transistors scaled to a 40-nanometer footprint," *Science*, vol. 356, no. 6345, pp. 1369–1372, 2017.

[55] J. Tang, Q. Cao, G. Tulevski, K. A. Jenkins, L. Nela, D. B. Farmer, and S.-J. Han, "Flexible cmos integrated circuits based on carbon nanotubes with sub-10 ns stage delays," *Nature Electronics*, vol. 1, no. 3, p. 191, 2018.

[56] Various, "Irds^{TM} 2016 edition white papers," 2017.





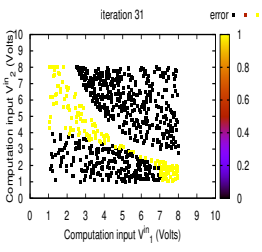
Chapter 3

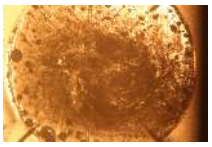
Problem Formulation and Hardware Platform

3.1 General Overview

This chapter provides a formal definition, in mathematical terms, of the classical evolution in materio (EiM) framework implemented in experiments, along with a description of the computational problems and evolutionary algorithms (EAs) used to solve them *in materio*. The hardware used to interface with the materials discussed in Chapter 2 is also presented.

Models are commonly used in experimental investigations. They can provide understanding of a system's mechanisms, whilst remaining inexpensive in terms of time, energy and cost. For example, deterministic models of memristive behaviour have been used to solve various computational tasks, either analytically [1] or through their integration into artificial neural networks (ANN) [2–4]. Using EAs to train a model of material is referred to as extrinsic evolution, or evolution *in silico*, as opposed to intrinsic evolution, or evolution *in materio*, where the material itself is trained. An example of extrinsic evolution was reported in [5]. In this case, a network of gold nanoparticles was modelled using an ANN obtained via empirical measurements. The fact that the model was based on experimental data, suggested that using the material itself would yield similar results [5, 6].





However, models are not always accurate. In order to avoid the potential loss of information that would result from inaccuracies, it was decided not to use models in this work. Instead, intrinsic evolution was performed. This is consistent with EiM's aim to enable EAs to explore potentially unknown, or indeed unknowable, properties of materials. In addition, to the author's knowledge, there is currently no model of the specific liquid SWCNT-based composites which are a primary subject of this thesis' investigations. It must also be noted that compared to areas of research where experiments are very expensive, time consuming and/or dangerous to run, EiM, as implemented here was not prohibitive in terms of cost, time and risk.

Artificial learning (AL) was used in order to solve the problem of transforming materials into devices capable of processing information. The choice of AL was motivated by its ability to solve problems that are not well defined, or for which no known solutions exist [7]. Discussions regarding the definitions and applications of the different AL approaches which are beyond the scope of this thesis are reported in [8–10]. Here, the EiM problem was solved using supervised learning, to modify EA-controlled inputs applied to the material and resulting in changes in the latter's electrical state. A simplistic representation of the implementation is presented in Figure 3.1.

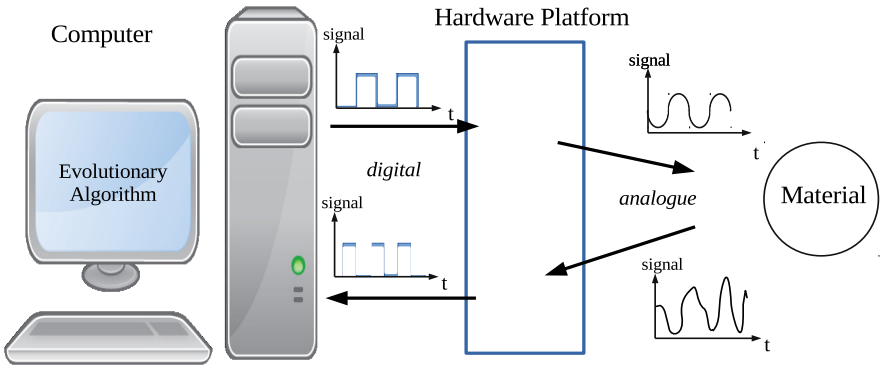
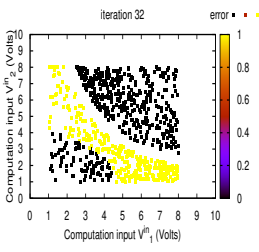
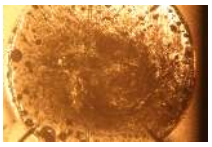


FIGURE 3.1: Basic implementation of EiM experiment. Signals produced by an EA are applied to a hardware platform where they are transformed into analogue signals applied to the material. The material's state is measured and the resulting signals are sent back to the EA.

This chapter is organised in two main parts; the first details the mathematical implementation of the classical EiM framework used in experiments, including problem formulation, computation problems and algorithms, the second presents the hardware used to interface with the material. A summary of the experimental implementation is provided in the last section.



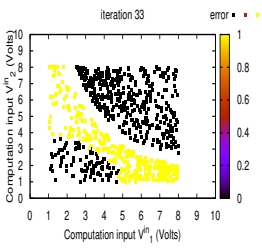


Important notation and definitions

Tables 3.1 and 3.2 present the notation and definition of the most important parameters, variables, indexes and results that will be introduced in this chapter and used subsequently. The aim of these two tables is to simplifying the reading of this chapter, since it is notation heavy. Exhaustive definitions are found in the text.

TABLE 3.1: Important parameters and variables of the problem formulation.

notation	definition	parameter	variable
n_1	number of *computation* input	✓	
n_2	number of *configuration* input	✓	
n_3	number of output measurements	✓	
n_4	number of additional problem variables	✓	
$\mathbf{V}^{\mathcal{C}}$	vector of *computation* inputs		✓
$\mathcal{C}()$	computation outcome, i.e. known class of inputs	✓	
$\mathbf{x}$	vector defining the state of a device	✓	✓
		material dependent	
$\mathbf{V}$	vector of *configuration* inputs		✓
$\mathbf{M}$	material state	✓	✓
		material dependent	
$\mathbf{R}$	vector of additional problem variables		✓
$\mathbf{x}'$	only the well defined quantities of $\mathbf{x}$, i.e. not $\mathbf{M}$		✓
$\mathbf{Y}(\mathbf{M})$	output measurements	✓	
$\mathcal{C}_M()$	computation outcome based on output measurements	✓	
$S_{\mathcal{C}}$	interpretation scheme ('translates' $\mathbf{Y}(\mathbf{M})$ into $\mathcal{C}_M()$)	✓	
T_1 and T_2	error threshold used in the termination criteria	✓	
K	number of input pair (attribute/class) in a dataset	✓	
Λ	maximum number of iterations	✓	
Q	maximum number of verification tests	✓	
N	population size of the EA(s)	✓	
D	number of EA-controlled dimensions ($= n_2 + n_4$)	✓	



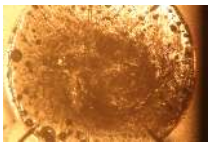


TABLE 3.2: Index definitions and important notations.

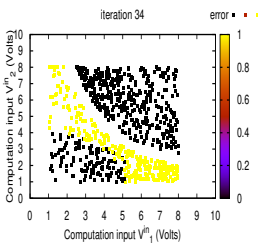
notation	definition
λ	iteration index
l	individual index
k	computation input/class pair (instance) index
t	indicates a belonging to the training dataset
v	indicates a belonging to the verification dataset
i	verification test index
$*$	indicates the best result achieved
$-$	indicates an average
K_t	number of training input/class pairs (instances)
K_v	number of verification input/class pairs (instances)
Φ_e^t	training error averaged over K_t
$\Phi_e^{v,i}$	verification error averaged over K_v, for one test i
$\Phi_e^{t,*}$	best error obtained during training (over all $\lambda \leq \Lambda$)
$\Phi_e^{v,*}$	best error obtained during verification tests (over all $i \leq Q$)
$\overline{\Phi_e^v}$	error obtained during verification tests averaged over Q

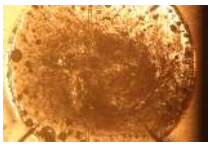
3.2 EiM as Optimisation Problem

The process of evolving the characteristics of a material such that it becomes able to solve a computational problem can be formulated as an optimisation problem. Finding the correct solution that produces a problem solving state in the material becomes the optimisation problem's objective. Different types of optimisation algorithms can be used to achieve this objective through a manipulation of the material's state. The use of optimisation as a way of implementing EiM is consistent across EiM investigations. However, the exact formulation of the optimisation problem varies according to the computational problem, materials and algorithms studied. The following subsections detail, in mathematical terms, the formulation used here.

3.2.1 Computation and Configuration Inputs

Two types of input signals exist in our implementation, *computation* and *configuration*. Due to the choice of material, both come in the form of direct current (DC) voltages applied on the material at selected electrodes.





Computation inputs: A simple definition of computation as a mapping $\mathcal{C}$ from a domain of definition $\mathcal{A}$ to a range of values $\mathcal{D}$, is adopted, which reads

$$\mathcal{C} : \mathcal{A} \rightarrow \mathcal{D}. \tag{3.1}$$

$\mathcal{C}$ receives $n_1 \in \mathbb{N}^+$ computation inputs which are uniquely mapped to a point in $\mathcal{D}$. The computation inputs are organised into vector

$$\mathbf{V}^{\mathcal{C}} \in \mathcal{A} \subset \mathbb{R}^{n_1} \tag{3.2}$$

and the unique computation outcome is

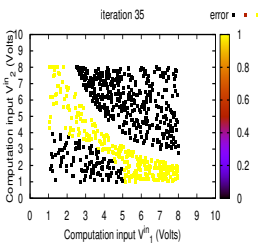
$$\mathcal{C}\left(\mathbf{V}^{\mathcal{C}}\right) \in \mathcal{D} \tag{3.3}$$

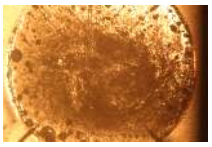
Since the property modified during training is a material's electrical input/output relationship, it was decided to apply the computation inputs as voltages to the material. They are therefore referred to as *computation input voltages*. When the computation inputs are converted into the computation input voltages they will need to be scaled to fit hardware and material constraints. For example, a computation input $V_1^{\mathcal{C}} = 90$ cannot simply be converted into an analogue voltage level with an amplitude of 90 V if the hardware has an input voltage limit at 20 V. In this hypothetical case, all computation inputs have to be scaled down such that the maximum computation input voltage is below 20 V. The domain of definition $\mathcal{A}$ effectively becomes a box defined by the inequalities

$$\mathcal{A} = \left\{ \mathbf{V}^{\mathcal{C}} \in \mathbb{R}^{n_1} : V_{i,\min}^{\mathcal{C}} \leq V_i^{\mathcal{C}} \leq V_{i,\max}^{\mathcal{C}},\ i = 1, \ldots, n_1 \right\} \tag{3.4}$$

where $V_{i,\min}^{\mathcal{C}}$ and $V_{i,\max}^{\mathcal{C}}$ are the minimum and maximum allowed values of the computation input voltages.

Configuration inputs: In order to change the material state in some desirable way, configuration inputs are realised as voltages when they are applied to the material through selected electrodes. They are therefore referred to as *configuration voltages*. These are independent from the computation inputs and their number $n_2 \in \mathbb{N}^+$ depends on the problem formulation and hardware capabilities. These variables govern the material's electrical and morphological evolution and constitute part of the training optimisation





problem's decision variables. They are organised into vector

$$\mathbf{V} \in \mathcal{B} \subset \mathbb{R}^{n_2} \tag{3.5}$$

where $\mathcal{B}$ is the box defined by the inequalities

$$\mathcal{B} = \left\{ \mathbf{V} \in \mathbb{R}^{n_2} : V_{i,\min} \leq V_i \leq V_{i,\max},\ i = 1, \ldots, n_2 \right\} \tag{3.6}$$

where $V_{i,\min}$ and $V_{i,\max}$ are the minimum and maximum allowed values of the configuration inputs and are known problem parameters whose level is constrained by the hardware.

3.2.2 Performing a Computation

EiM's objective is to bring a material sample to a state $\mathbf{M}$ such that when computation inputs $\mathbf{V}^{\mathcal{C}}$ are applied to it, its response can be interpreted as a pre-defined computation. This response has the form of $n_3 \in \mathbb{N}^+$ measurements selected from a range of possible physical quantities characterising the material, and organised into vector $\mathbf{Y}\left(\mathbf{M}\right) \in \mathbb{R}^{n_3}$.

Let $S_{\mathcal{C}}$ be an *interpretation scheme* used for converting the material's response into a computation $\mathcal{C}$. When $S_{\mathcal{C}}$ is used for a particular set of computation inputs $\mathbf{V}^{\mathcal{C}}$ applied on the material in state $\mathbf{M}$, resulting in response $\mathbf{Y}\left(\mathbf{M}\right)$, while configuration inputs $\mathbf{V}$ are applied, the outcome is a unique value

$$S_{\mathcal{C}}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\left(\mathbf{M}\right), \mathbf{R}\right) \in \mathcal{D} \tag{3.7}$$

where $\mathbf{R} \in \mathbb{R}^{n_4}$ is an n_4-dimensional vector of problem dependent quantities, included in the analytical functional expression of $S_{\mathcal{C}}$. $\mathbf{R}$ can be treated as:

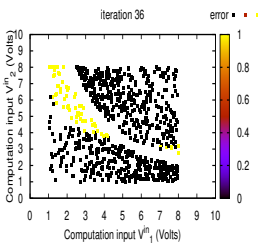
(a) a set of parameters with *a priori* known values, or

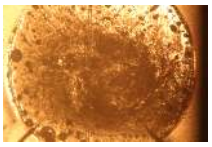
(b) independent decision variables of the optimisation problem, or

(c) quantities with a pre-specified functional dependence $\mathbf{G}$ on the configuration input, i.e. as $\mathbf{R} = \mathbf{G}\left(\mathbf{V}\right)$.

In case $\mathbf{R}$ is considered a vector of decision variables, such that $\mathbf{R} = [R_1 \ldots R_{n_4}]^T$, it is bounded bounded within a box

$$\mathcal{R} = \left\{ \mathbf{R} \in \mathbb{R}^{n_4} : R_{i,\min} \leq R_i \leq R_{i,\max}, i = 1, \ldots, n_4 \right\} \tag{3.8}$$





where $R_{i,\min}$ and $R_{i,\max}$ are lower and upper bounds of each decision variable R_i, respectively.

The dependence of $S_{\mathcal{C}}$ on $\mathbf{M}$ cannot be considered explicitly due to the complexity of the random material morphology. It is considered implicitly through the measurement vector $\mathbf{Y}(\mathbf{M})$. $\mathbf{M}$ is changed over time by the combined effect of the repeated application of computation and configuration inputs. It is a function of the trajectories of $\mathbf{V}^{\mathcal{C}}_{(\lambda,\ell)} \in \mathcal{A}$ and $\mathbf{V}_{(\lambda,\ell)} \in \mathcal{B}$, where λ is the iteration index of an evolutionary population-based optimisation algorithm and ℓ the index of the individuals in that population. The calculation of the error for ℓ involves the application of the configuration inputs $\mathbf{V}_{(\lambda,\ell)}$ as well as the application of all the computation inputs in the training dataset.

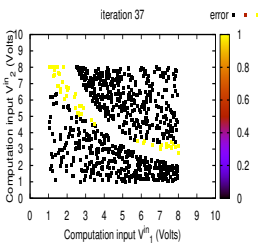
The training problem in EiM is to modify the material state so that the application of $S_{\mathcal{C}}$ for the whole range of possible computation inputs from $\mathcal{A}$ yields the correct calculation according to specifications (3.1)–(3.3).

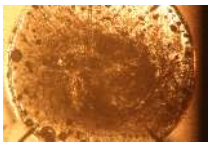
3.2.3 EiM Training Problem Formulation

In order to bring the material into a computation inducing state, its electrical input/output relationship is modified within an optimisation loop that aims at minimising a measure of the computation error. Since the material is effectively treated as a black box, and the exact material behaviour is unknown, a supervised learning approach is followed [11, 12]. An experiment is split into a training phase where a solution to the training problem is produced, and a verification phase, where the solution is tested against new data.

Let $\mathcal{V}^{\mathcal{C}}_t$ be the training dataset used for computation problem $\mathcal{C}$ consisting of K_t pairs $\left(\mathbf{V}^{\mathcal{C}}_t(k), \mathcal{C}\left(\mathbf{V}^{\mathcal{C}}_t(k)\right)\right)$, $k = 1, \ldots, K_t$, satisfying (3.1)–(3.3), where t denotes belonging to the training phase. According to (3.7), the application of computation input $\mathbf{V}^{\mathcal{C}}_t(k)$, along with a set of configuration inputs $\mathbf{V}$, leads to the computation $S_{\mathcal{C}}\left(\mathbf{V}^{\mathcal{C}}_t(k), \mathbf{V}, \mathbf{Y}(\mathbf{M})(k), \mathbf{R}\right) \in \mathcal{D}$. The resulting value can be compared to the $\mathcal{C}\left(\mathbf{V}^{\mathcal{C}}_t(k)\right)$ corresponding to the specific $\mathbf{V}^{\mathcal{C}}_t(k)$ applied to the material. This comparison allows the definition of a training measure of error.

Now, let us define the device state $\mathbf{x}$ as the vector that contains the information regarding the configuration input $\mathbf{V}$, the material state $\mathbf{M}$ and the auxiliary quantities $\mathbf{R}$. Depending on the policy followed for $\mathbf{R}$, three different state vectors can be defined





with $\mathbf{R}$ excluded, explicitly included, or implicitly included, corresponding to options (a),(b),(c) outlined in section 3.2.2. For the sake of brevity and consistency, $\mathbf{x}$ is defined as

$$\mathbf{x} = [\mathbf{V}\ \mathbf{R}\ \mathbf{M}]^T . \tag{3.9}$$

$\mathbf{V}$ and $\mathbf{R}$ are well defined quantities in $\mathcal{B}$ and $\mathcal{R}$, respectively. $\mathbf{M}$, as previously discussed, is not well-defined since it consists of all possible states of the material. Thus, $\mathbf{x}$ is split into two corresponding components, $\mathbf{x} = \left[\mathbf{x}'\ \mathbf{M}\right]^T$ with $\mathbf{x}' = [\mathbf{V}\ \mathbf{R}]^T$.

In the case of SWCNT-based composites, $\mathbf{M}$ reflects the arrangements of all nanotubes within the matrix in which they are dispersed, forming percolation paths of variable electrical conductivity. In dynamic SWCNT-based composites, $\mathbf{M}$ belongs to the intractable search space of possible SWCNT network realisations in three dimensions, in the confined space of the container, where the material is drop-cast. In the solid composites, $\mathbf{M}$ does not belong to the set of decision variable but is instead a problem parameter. In this case, $\mathbf{M}$ is specific to the sample used, since the network of SWCNTs, achieved when the material has solidified, remains the same throughout experiments.

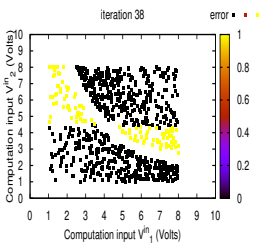
Because of the dependence of $\mathbf{M}$ on consecutive applications of computation and configuration inputs, the feedback mechanism based on $\mathbf{Y}$ $(\mathbf{M})$ and $S_{\mathcal{C}}$ allows the definition of an optimisation problem with $\mathbf{x}$ as the vector of decision variables and the hardware within the loop. The objective function during the material's training phase is a measure of the total error over $\mathcal{V}_t^{\mathcal{C}}$.

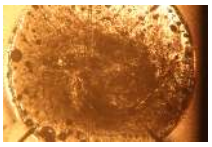
The device's computation error $\epsilon_{\mathbf{x}}$ when it is in state $\mathbf{x}$ and the training computation input vector $\mathbf{V}_t^{\mathcal{C}}(k)$ is applied, along with $\mathbf{V}$, is

$$\epsilon_{\mathbf{x}}\left(\mathbf{V}_t^{\mathcal{C}}(k)\right) = g\left\{S_{\mathcal{C}}\left(\mathbf{V}_t^{\mathcal{C}}(k), \mathbf{V}, \mathbf{Y}(k), \mathbf{R}\right), \mathcal{C}\left(\mathbf{V}_t^{\mathcal{C}}(k)\right)\right\} \tag{3.10}$$

where g is a suitably pre-selected error function. For the sake of simplicity, this equation does not report the dependence of $\mathbf{Y}$ on $\mathbf{M}$ as it is implicit. Thus, the mean training error over $\mathcal{V}_t^{\mathcal{C}}$ for device state $\mathbf{x}$ is

$$\Phi_e\left(\mathbf{x}, \mathbf{V}_t^{\mathcal{C}}, K_t\right) = \frac{1}{K_t}\sum_{k=1}^{K_t}\epsilon_{\mathbf{x}}\left(\mathbf{V}_t^{\mathcal{C}}(k)\right) = \Phi_e^t \tag{3.11}$$





and the EiM training optimisation problem is expressed as

$$\min_{\mathbf{x}} \Phi_e \left(\mathbf{x}, \mathbf{V}_t^{\mathcal{C}}, K_t \right) \tag{3.12}$$

subject to

$$\mathbf{V} \in \mathcal{B} \tag{3.13}$$

$$\mathbf{R} \in \mathcal{R} \tag{3.14}$$

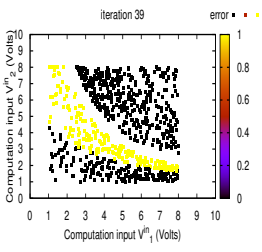
$$\mathbf{M} \text{ is a feasible material state.} \tag{3.15}$$

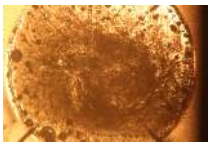
An EA solving optimisation problem (3.12)–(3.15) cannot directly assign a material configuration defining $\mathbf{M}$. Instead, it can iteratively drive the material towards forming internal structures (liquid samples), or find existing ones (solid samples), both cases favouring minimisation of the computation error by manipulating $\mathbf{V}$ and $\mathbf{R}$.

$\mathbf{M}$ is a representation of the real material and since it can only assume feasible states in the hardware implementation, constraint (3.15) can be neglected. A computer-based simulation of the material behaviour replacing the physical matter would require a mechanism for the explicit consideration of (3.15) for assuring feasibility. The training optimisation problem (3.12)–(3.14) is solved using an EA that converges to an optimal point $\mathbf{x}^* = [\mathbf{V}^* \ \mathbf{R}^* \ \mathbf{M}^*]^T$.

Defining termination conditions for this algorithm is difficult because even if $\mathbf{V}$ and $\mathbf{R}$ are trapped within a basin of attraction, $\mathbf{M}$ will still be changing due to the repeated application of $\mathbf{V}_t^{\mathcal{C}}$ and $\mathbf{V}$. Assuming no charge trapping mechanism in the solid materials, these changes only apply to the liquids materials. In the latter case, it is possible that changes produced by the application of the computation and configuration inputs are irreversible. A material drifting effect is therefore inevitable. However, even in this case, a notion of material convergence can be observed, in the sense that if the training is successful, the progressive build-up of internal structures is robust enough to result in the desired computation inducing state.

In order to define termination conditions, let Λ denote the maximum number of iterations. Considering a population based algorithm with population size N, used for solving problem (3.12)–(3.14), let $\Phi_e^{t,(\lambda,\ell)}$ be the value of the objective function (3.11)





of individual $\ell \in N$ at iteration λ. The average iteration error is

$$\overline{\Phi_e^{t,\lambda}} = \frac{1}{N}\sum_{\ell=1}^{N} \Phi_e^{t,(\lambda,\ell)} \tag{3.16}$$

and (λ, ℓ^*) is the individual achieving the minimum error at iteration λ, i.e.

$$\left(\lambda, \ell^*\right) = \underset{\ell \in 1,\ldots,N}{\arg\min} \quad \Phi_e^{t,(\lambda,\ell)}. \tag{3.17}$$

The algorithm terminates when

$$\textbf{condition 1: } \lambda = \Lambda \text{ or} \tag{3.18}$$

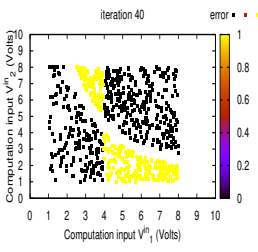
$$\textbf{condition 2: } \Phi_e^{t,(\lambda,\ell^*)} \leq T_1 \ \wedge \ \left(\overline{\Phi_e^{t,\lambda}} - \Phi_e^{t,(\lambda,\ell^*)}\right) \leq T_2 \tag{3.19}$$

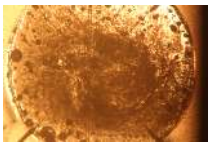
where T_1 and T_2 are preselected error threshold values. The rationale behind the use of these two termination conditions differs depending on whether the material is solid or liquid.

In the case of a solid material, it is time-consuming to let the process continue if a solution within the error threshold values has been found. Irrespective of the termination condition, differences in classification error between training and verification will be an indication of the quality of the solution $\mathbf{x}'^*$, and its ability to induce a computing state in the material given $\mathbf{M}$. Condition 2 is therefore preferable to condition 1 as it ensures that a good solution is always found before termination of the training phase. This can result in reduced training time, but it can also result in the reverse. If no solution satisfying condition 2 exists in the material, then training will run forever. This is why both condition 1 and condition 2 are used in the problem formulation.

In the case of the liquid samples, it is also expected that if the training has resulted to a material state that performs well, then sufficiently small minimum and average iteration errors are good indications of material convergence. In this case however, it is possible that the solution where the best result was achieved cannot be fully recovered at the end of the training phase, due to material drift. The nature of the solution selected to be used for the verification phase therefore depends on which of the two termination conditions was fulfilled.

If the algorithm ended due to termination condition 1, the optimal solution $\mathbf{x}^* =$





$\left[\mathbf{x}'^{*}\ \mathbf{M}^{*}\right]^{T}$ is selected from the recorded history of all points $\mathbf{x}^{(\lambda,\ell)}$ visited by the algorithm and the corresponding error value $\Phi_e^{t,(\lambda,\ell)}$. The $\mathbf{x}'^{*}$ part of the solution is selected from iteration λ^* and individual ℓ^*, which yielded the minimum error $\Phi_e^{t,(\lambda^*,\ell^*)}$, simplified to $\Phi_e^{t,*}$, and is given from

$$\left(\lambda^*, \ell^*\right) = \underset{\substack{(\lambda,\ell)\\ \lambda=1,\ldots,\Lambda\ ,\ell=1,\ldots,N}}{\arg\min} \Phi_e^{t,(\lambda,\ell)}. \tag{3.20}$$

The $\mathbf{M}^*$ part of the solution used, however, is $\mathbf{M}^{(\Lambda,N)}$, i.e. the material state after the last evaluation of the objective function from the population's final individual. $\mathbf{M}^{(\Lambda,N)}$ and $\mathbf{M}^*$ can coincide if $\mathbf{x}^*$ was achieved when $\lambda^* = \Lambda$ and $\ell^* = N$. If this is not the case, $\mathbf{M}^{(\Lambda,N)}$ is probably different from $\mathbf{M}^*$, which means that the $\Phi_e^{t,*}$ is no longer representative of the quality of the evolved device subjected to $\mathbf{x}'^{*}$.

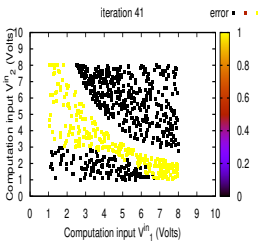
If the algorithm terminated due to condition 2, then

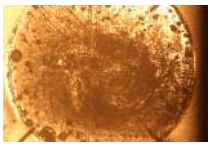
$$\left(\lambda^*, \ell^*\right) = \left(\lambda, \ell^*\right) \tag{3.21}$$

and $\mathbf{M}^* = \mathbf{M}^{(\lambda^*,N)}$, i.e. the material state following the last objective function from the population's final individual at the iteration where condition 2 was satisfied. Termination due to condition 2 is preferable to condition 1, as for solid materials, but in this case it is due to the fact that condition 2 reduces the number of function evaluations and therefore irreversible changes in the material after a good enough solution has been found. In case of condition 1, this number is $\left[N - \ell^* + N(\Lambda - \lambda^*)\right]$ whereas in case of condition 2 it is $(N - \ell^*)$. The fewer function evaluations from λ^* to the algorithm's termination, the better the quality of the solution used in the verification phase. Both conditions are used, however, since it is possible that no solution satisfying condition 2 is found within a reasonable time-frame, justifying the need for condition 1.

3.2.4 EiM Solution Verification

The quality of $\mathbf{x}^*$ is evaluated by considering a verification dataset $\mathcal{V}_v^{\mathcal{C}}$, which consists of $K_v > K_t$ pairs $\left(\mathbf{V}_v^{\mathcal{C}}(k), \mathcal{C}\left(\mathbf{V}_v^{\mathcal{C}}(k)\right)\right)$, with $\left|\mathcal{V}_t^{\mathcal{C}} \cap \mathcal{V}_v^{\mathcal{C}}\right|$ small if not zero.





By applying the optimal configuration inputs $\mathbf{V}^*$, to a material brought to the optimal state $\mathbf{M}^*$ and using the optimal parameter set $\mathbf{R}^*$, all computation inputs of $\mathcal{V}_v^{\mathcal{C}}$ are sent to the material and the corresponding responses $\mathbf{Y}(k)$ are recorded. Based on these recordings, the interpretation scheme (3.7) is applied and the verification error calculated is $\Phi_e\left(\mathbf{x}^*, \mathbf{V}_v^{\mathcal{C}}, K_v\right)$.

In view of the material drift effect, a good training solution should result in a relatively robust material structure retaining the good properties of $\mathbf{M}^*$ when $\mathbf{x}^*$ was obtained. A single calculation of $\Phi_e^{v,(\lambda^*,\ell^*)}$ is actually the mean value of Q repeated verification tests using $\mathcal{V}_v^{\mathcal{C}}$ in Q separate runs on the trained material, i.e.

$$\Phi_e^v\left(\mathbf{x}^*, \mathbf{V}_v^{\mathcal{C}}, K_v\right) = \frac{1}{Q}\sum_{i=1}^{Q}\Phi_e^{v,i}\left(\mathbf{x}^*, \mathbf{V}_v^{\mathcal{C}}, K_v\right) \tag{3.22}$$

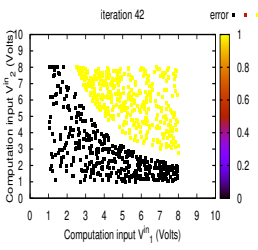
where $\Phi_e^{v,i}\left(\mathbf{x}^*, \mathbf{V}_v^{\mathcal{C}}, K_v\right)$ is the error of verification trial i of the same solution and material immediately after training.

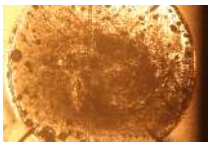
3.3 Computational Problems for EiM

A number of computational problems have been considered for EiM investigations. A comprehensive list is presented in [13]. This list is based on observations reported in [14–16]. The suitability of each problem is assessed in terms of its general interest for the community, such as whether it is it hard to solve using conventional methods, and in terms of the potential for EiM to solve it. From this list, data classification and the Exclusive-**OR** (**XOR**) logic gate were chosen for the investigations presented here. Classification is described in the following sections, whilst the **XOR** problem will be detailed in Chapter 8 where it is implemented in experiments.

3.3.1 Data Classification

Why choose classification as a problem for EiM? Classification is a tool commonly used in data analysis and decision making, where data is categorised according to common features or other pre-defined discriminating conditions. A perfect classifier can be constructed given complete knowledge of the relationship between available data and its class. However, this knowledge is not always available, for example when a dataset's





scale and complexity are high. With the rise in data availability and interest for its analysis, finding ways of producing fast and accurate classifiers has become the focus of number of investigations within both conventional and unconventional computing communities.

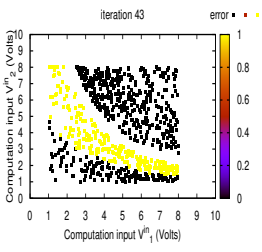
Supervised learning is one of the methods studied for solving complex classification problems [10]. Classifiers produced using this method to train artificial machines running *in silico* such as ANNs have been successfully applied to a variety of problems [9]. However, it has been observed that a very large number of training data is generally necessary to obtain good solutions, and the resulting classifiers have a tendency to over-fit, i.e. they do not classify well new or unseen data instances. This poses problems, as large training datasets are not always available and in critical cases, such as medical and engineering applications, over-fitting can have disastrous consequences. Other methods are being developed successfully, however, room for improvement remains, motivating the study of classification problems for EiM.

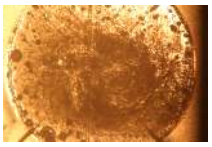
In addition to the interest of classification within the computing community and the number of fields it can be applied to, this type of problem has already been investigated with solid SWCNT/polymer samples. SWCNT/poly(methyl meta-acrylate) (PMMA) and SWCNT/poly(buthyl meta-acrylate) (PBMA) classifiers have been evolved using both the classical EiM [17, 18] and the RCiM frameworks [19]. In general, the classification problems, such as Lenses [20] or Iris [21], were retrieved from the UCI repository [22]. In the case of the Iris dataset, it was observed that the RCiM produced solutions that were consistently more accurate than those obtained with EiM, but slightly worst than cartesian genetic program (CGP) or ANN run *in silico*.

These observations suggest that 1) classification problems have the potential to be solved using EiM, 2) modifying the implementation can improve results and 3) investigations are still needed to establish whether EiM-produced classifiers can be competitive alternatives to silicon-based ones. Finally, the previous EiM investigations provide means for comparison with the implementation used here.

Data classification problem description

In the simple definition of a computation as a top level input/output process, the input is the vector of characteristic features and the output is the classification result. In this case,





n_1 is the number of characteristic features of a particular dataset organised in $\mathbf{V}^{\mathcal{C}}$. They can be continuous, discrete or even qualitative. Two basic approaches for addressing classification problems exist: generative and discriminative [23].

Generative methods model the joint probability distributions of the data source inputs and outputs, allowing synthetic data to be generated. Hence, there is an *inference* problem concerned with the calculation of the probability that a $\mathbf{V}^{\mathcal{C}}$ belongs to class $\mathcal{A}_i$, and a *decision* problem that assigns it to one of the possible classes based on a decision theoretic approach.

Discriminative methods do not consider these probabilities, but instead try to identify a *discriminant function* $f\left(\mathbf{V}^{\mathcal{C}}\right)$, which maps directly $\mathbf{V}^{\mathcal{C}}$ to one of the classes $\mathcal{A}_i$, [23]. In this sense, the EiM approach replaces the explicit definition of a classification discriminant function with a material sample. This sample is trained such that when incident signals $\mathbf{V}^{\mathcal{C}}$ are applied to it, the response measured is interpreted as a unique class assignment.

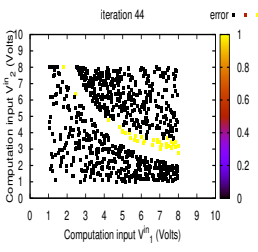
A classification problem with a number L of classes has $\mathcal{D} = \{1, \ldots, L\}$. In this case, $\mathcal{A}$, consists of L subspaces $\mathcal{A}_i$, $i = 1, \ldots, L$ which correspond to classes $1, \ldots L$, with $\mathcal{A}_1 \cup \mathcal{A}_2, \ldots, \mathcal{A}_{L-1} \cup \mathcal{A}_L = \mathcal{A}$. In the case of fully separable classes $\mathcal{A}_1 \cap \mathcal{A}_2, \ldots, \cap \mathcal{A}_L = \emptyset$, whereas for partially overlapping classes, at least $\mathcal{A}_i \cap \mathcal{A}_j \neq \emptyset$, where $i \neq j$. The classifier is given computation inputs $\mathbf{V}^{\mathcal{C}} \in \mathcal{A}$ and assigns them to a class i. In this sense the computation to be performed by the evolved classifier is

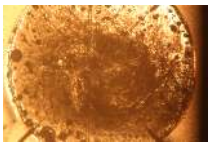
$$\mathcal{C}\left(\mathbf{V}^{\mathcal{C}}\right) = i \quad \text{if} \quad \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_i \ . \tag{3.23}$$

3.4 Evolutionary Algorithms

3.4.1 General Characteristics

When following the EiM implementation presented so far, training algorithms have two ways of solving the optimisation problem, thereby providing a solution to the computation problem. If the solid materials (memristor, SWCNT/PBMA, resistor) are used, the search space explored by the algorithm consists in the set of configuration inputs influencing the state of these materials. On the other hand, when liquid samples (SWCNT/LC, SWCNT/epoxy, microtubules) are used, the algorithm searches a hybrid space of solutions, which consists of 1) a subspace spanned by the configuration inputs or stimuli





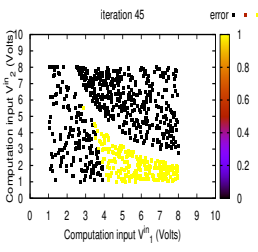
influencing the material state and 2) a subspace formed by a network of SWCNT bundles (or microtubules) within the liquid matrix [24–26]. With liquid samples, the space of possible network configurations and associated percolation paths is infinite dimensional and dynamic, as it can change at every step. With solid samples, the space of network configurations is finite and dependent on the structures resulting from the material's preparation.

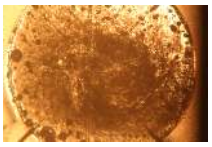
In both cases, the search algorithm has only implicit access to the space and the subspaces through the configuration inputs. The lack of an analytical model of the material's electrical behaviour, added to the non-linear, dynamic and near-chaotic [27] nature of the search space, directed the choice of algorithms towards stochastic and derivative-free optimisation algorithms. EAs present such characteristics, and have demonstrated their capacity to find solutions in this type of search space, motivating their use as search algorithms.

EAs used in EiM investigations include genetic algorithms (GA), used to solve various computational problems in liquid crystal (LC) [15, 28–31] and solid SWCNT/polymer composites [32–36]. The composites were also evolved using evolutionary strategies (ES) [17, 37–39], Nelder-Mead (NM) [40], differential evolution (DE), and particle swarm optimisation (PSO) [18, 41]. Other algorithms were proposed in [42] for the Nascence project but have not been used in published results.

Investigations into the effect of different EA characteristics on experimental results have compared NM with DE [43]. The problem was that of performing Boolean logic in solid SWCNT/polymer samples. It was observed that NM was less consistent than DE with respect to accuracy. Other investigations include comparison between DE and PSO's search in SWCNT/LC samples [44, 45]. However, a more exhaustive comparative study would be needed before a learned suggestion of optimal algorithm/material/problem combinations can be reported.

Investigations reported here were primarily undertaken with DE and PSO. Both algorithms have been applied successfully to solve a variety of computational problems and have been the subject of extensive theoretical [46, 47] and experimental research [48, 49]. As a result, a large reference library providing potential variants on their basic implementations exists. In addition, DE and PSO have often been compared in terms of results, convergence rate, computational efficiency [50]. DE tends to have a slower





convergence, but is generally more robust and less prone to getting stuck in local minima. PSO search tends to focus on exploration rather than exploitation, enabling it to converge faster to potential optima.

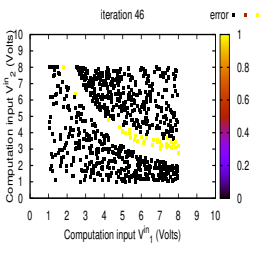
Both algorithms have also been used in the context of EiM, to solve a variety of computational problems in SWCNT/PBMA, with results comparable to genetic algorithm (GA) and evolution strategies (ES) [18]. However, DE and PSO have not been implemented as much as GA and ES, or used in many other materials, leaving avenue for research and discovery. Finally, DE and PSO are very versatile, in the sense that their parameters and structure can easily be customised to suit a particular problem. For the sake of simplicity, the notation used in this work to denote the same concepts, such as population or individual, will be kept the same for both algorithms, despite potential differences in the names used in literature to denote these concepts.

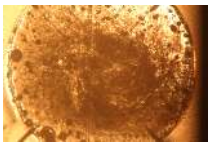
3.4.2 Differential Evolution

DE is a heuristic search algorithm proposed by Storn and Price in 1996 [51]. It is able to solve optimisation problems with non-differentiable and non-linear objective functions. In this algorithm, at each iteration $\lambda \in \Lambda$, a population contains N individuals $\mathbf{x}^{(\lambda,\ell)}$, $\ell = 1, \ldots, N$. Each individual corresponds to a vector of decision variables with dimension $d \in D$. A decision variable is denoted $\mathbf{x}_d^{(\lambda,\ell)}, d = 1, \ldots, D$.

Individuals represent potential solutions to an optimisation problem where an objective, or fitness function, Φ_e^t must be optimised. The problem's solution is optimal vector $\mathbf{x}^*$ producing $\Phi^* = \Phi_e^t(\mathbf{x}^*)$ which is the problem's (known) optimum. In order to find $\mathbf{x}^*$, all individuals are updated for a number of iterations. At a given iteration λ, an individual is subjected to a sequence of *mutations* and *cross-overs* to produce a test vector $\mathbf{x}^t$. The objective function is used to evaluate the original $\Phi(\mathbf{x}^{(\lambda,\ell)})$ and test vector $\Phi(\mathbf{x}^t)$. A *selection* rule based on their respective fitness, i.e. how close they are to the optimum Φ^* is then applied to choose which vector will be part of the population at iteration $\lambda + 1$. Algorithm 1 illustrates this evolutionary process.

Whilst the algorithm's structure is similar to the GA, the main operations: *mutation, cross-over* and *selection* differ in their form. At every iteration λ (or generation in GA terms), three vectors, $\mathbf{x}^{(\lambda,a)}, \mathbf{x}^{(\lambda,b)}, \mathbf{x}^{(\lambda,c)}$ are randomly drawn from the population, such





Algorithm 1 Differential Evolution (DE)

initialise parameters
iteration $\lambda = 0$
for each vectors (individuals) $\mathbf{x}^{(\lambda,\ell)}$, $\ell = 1, ...N$ **do**
    for all dimensions $d \in D$ **do**
        initialise $x_d^{(\lambda,\ell)} \sim U(x_{d,max} - x_{d,min})$
    evaluate vector using objective function, $\Phi(\mathbf{x}^{(\lambda,\ell)})$
    if $\Phi(\mathbf{x}^{(\lambda,\ell)}) = \Phi^*$ **then**
        solution has been reached, stop algorithm
while termination conditions not reached **do**
    for each $\mathbf{x}^{(\lambda,\ell)}$, $\ell = 1, ...N$ **do**
        produce test vector $\mathbf{x}^t$ using *mutation* and *cross-over*
        if $\Phi(\mathbf{x}^t) = \Phi^*$ **then**
            solution has been reached, stop algorithm
        else
            vector *selected* is passed on to next generation, $\mathbf{x}^{(\lambda+1,\ell)}$
    iteration = $\lambda + 1$

that $\mathbf{x}^{(\lambda,a)} \neq \mathbf{x}^{(\lambda,b)} \neq \mathbf{x}^{(\lambda,c)} \neq \mathbf{x}^{(\lambda,\ell)}$. The three vectors are then used to create a temporary test vector $\mathbf{x}^{t'}$ following the mutation operation,

$$\mathbf{x}^{t'} = \mathbf{x}^{(\lambda,a)} + F(\mathbf{x}^{(\lambda,b)} - \mathbf{x}^{(\lambda,c)}) \tag{3.24}$$

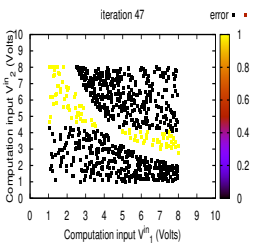
where F is the mutation parameter which controls the exploration of DE through the search space [52]. For each dimension, d, the cross-over operation is applied between $x^{t'}$ created using eq. (3.24) and the original vector $x^{(\lambda,\ell)}$,

$$x_d^t = \begin{cases} x_d^{t'} & \text{if } d = D \text{ or } r_d < CR \\ x_d^{(\lambda,\ell)} & \text{otherwise.} \end{cases} \tag{3.25}$$

where CR is the cross-over operator influencing the diversity of DE [52]. Greedy selection is then used, meaning that the vector best solving the problem is always chosen to be part of the next generation of solutions. In the case of a minimisation problem, given in eq.(3.15), the best solutions translate into lowest objective function:

$$\mathbf{x}^{(\lambda+1)} = \begin{cases} \mathbf{x}^t & \text{if } \Phi(\mathbf{x}^t) \leq \Phi(\mathbf{x}^{(\lambda,\ell)}) \\ \mathbf{x}^{(\lambda,\ell)} & \text{if } \Phi(\mathbf{x}^t) > \Phi(\mathbf{x}^{(\lambda,\ell)}) \end{cases} \tag{3.26}$$

Figure 3.2 illustrates the update of one individual $\mathbf{x}^{(\lambda,\ell)}$, $\ell = 1, ..., N$ from the population at iteration $\lambda \in \Lambda$, described in Algorithm 1 and following the rules presented in eqs. (3.24)-(3.26). For clarity, the number of dimensions is limited to two, i.e. $D = 2$.



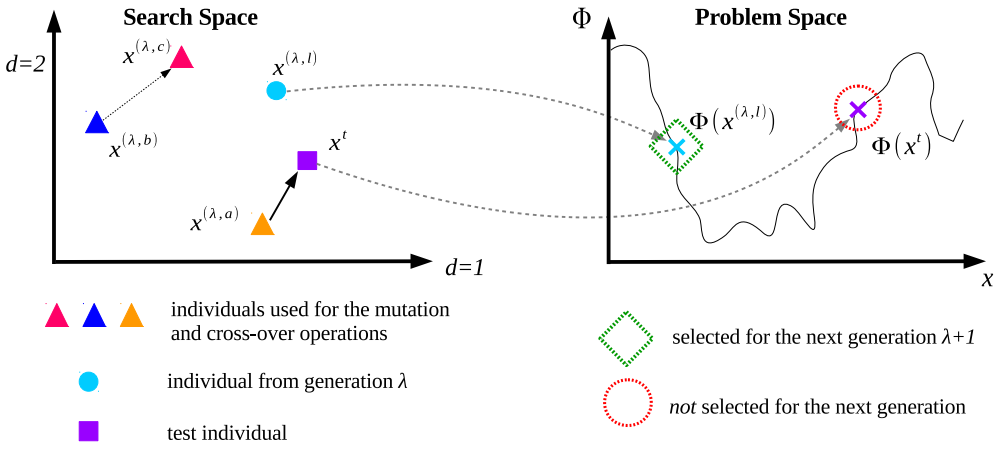
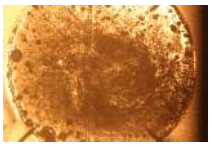


FIGURE 3.2: Update of an individual $\mathbf{x}^{(\lambda,l)}$ in a 2D search space using mutation and cross-over operations. The fitness of the test individual, $\Phi(\mathbf{x}^t)$, evaluated using the objective function, is worst than that of the original individual. The latter is therefore added to the next generation population.

DE variants generally present modified versions of at least one of the three operations used to update the population. A variant enabling more diversity in the population was first proposed in [51]. This variant uses four vectors in the weighted differential, which is added to the best individual, rather than a random one, to create the test vector. However, the benefits arise only for large populations, which would be impractical for the experiments undertaken with the liquid and solid SWCNT-based composites, where the time taken to evaluate the objective function is non negligible.

A review of other variants is presented in [53]. However, it is highlighted that little theoretical study regarding the convergence of these variants exist. Most observations are based on empirical studies and thus are implementation and problem dependent. A similar observation is made in [54] where it is added that variants tend to complicate the formulation, justifying this investigation's focus on parameter value instead.

As a result, the original version of DE was implemented here. In order to test the quality of the DE code developed for the EiM experiments, it was first implemented with the parameters suggested in [54] and tested against three benchmark optimisation functions: Rosenbrock, Rastrigin and Ackley. The results are presented in Appendix B, and were comparable to those obtained in [54]. This shows that the DE used in experiments compares with other DEs found in literature. However, due to the time taken for each solution evaluation in the case of EiM, the population size is $S = 8$, which is smaller than in most implementations found in literature. In addition, the choice of values for the differential weight $F = 0.814$ and cross-over operator $CR = 0.7026$, originally based



on [54] were modified following empirical investigations to fit the specificities of problem and implementation at hand. These DE parameters were used in all experiments, except otherwise stated.

3.4.3 Particle Swarm Optimisation

The concept and implementation of the particle swarm optimisation (PSO) algorithm was first discussed by Kennedy and Eberhart in 1995 [55]. PSO is classified as a swarm intelligence algorithm (SIA), which is sometimes distinguished from EAs. However, in this work, the term EA is used in its broadest sense: as a derivative-free iterative optimisation algorithm where solutions to optimisation problems are found by subjecting a population of solutions (which can be of one or more) to a series of modifications before selecting solutions for the next iteration, following a given selection rule. The term EA is therefore used to refer to both the DE algorithm and the PSO algorithm.

PSO is population-based, stochastic and derivative-free. It takes inspiration from the study of bird flocking behaviour. When implemented artificially, each bird becomes a particle and the flock is a group of N potential solutions. At iteration λ, a particle $\mathbf{x}^{(\lambda,\ell)}$, $\ell = 1, \ldots, N$ is defined by its current position $\mathbf{x}^{(\lambda,\ell)}$, ie: the vector of decision variables, following notation from section 3.2. It is also defined by its velocity $\zeta^{(\lambda,\ell)}$ and the past position which has achieved best fitness when evaluated using the objective function, the personal best: $\mathbf{x}^{b,(\lambda,\ell)}$. Each particle's behaviour depends on the way information regarding itself and others in the swarm is exchanged. Algorithm 2 details the update of a swarm of particles where information is exchanged globally [56].

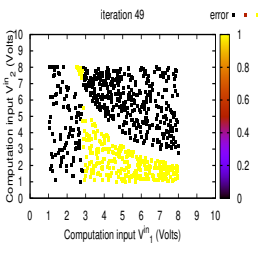
For the global PSO (GPSO) [57, 58] presented in Algorithm 2, over a number of dimensions D, the simplest update for a particle's velocity $\zeta_d^{(\lambda,\ell)}$, $d = 1, \ldots D$ at iteration λ is

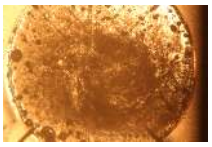
$$\zeta_d^{(\lambda+1,\ell)} = \zeta_d^{(\lambda,\ell)} + c_1 r_1 (x_d^{b,(\lambda,\ell)} - x_d^{(\lambda,\ell)}) + c_2 r_2 (x_d^{g,(\lambda)} - x_d^{(\lambda,\ell)}) \tag{3.27}$$

and the position $x_d^{(\lambda,\ell)}$ is given by

$$x_d^{\lambda+1,\ell} = x_d^{\lambda,\ell} + \zeta_d^{\lambda+1,\ell} \tag{3.28}$$

where $\mathbf{x}^{g,(\lambda)} = [x_1^{g,(\lambda)}, ..., x_D^{g,(\lambda)}]^T$ is the best vector of decision variables achieved by





the algorithm so far and evaluated over the whole swarm. The constant coefficients c_1, c_2 affect the size of the steps taken by the particle at every iteration and r_1, r_2 are factors adding stochasticity to the process. In the original paper [55], $c_1 = c_2 = 2$ and $r_1, r_2 \sim U(0, 1)$.

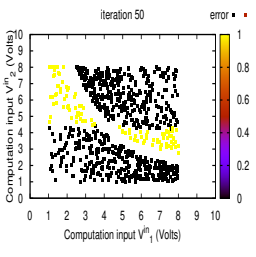
Algorithm 2 Global Particle Swarm Optimisation (GPSO)

initialise parameters
iteration $\lambda = 0$
for each particle $\mathbf{x}^{(\lambda,\ell)}, \ell = 1, \dots N$ **do**
    for all dimensions $d \in D$ **do**
        initialise $x_d^{(\lambda,\ell)} \sim U(x_{d,max} - x_{d,min})$
    personal best $\mathbf{x}^{b,(\lambda,\ell)} = \mathbf{x}^{(\lambda,\ell)}$
    evaluate particle using objective function, $\Phi(\mathbf{x}^{(\lambda,\ell)})$
    global best $\mathbf{x}^{g,(\lambda)} = \underset{\ell \in N}{\operatorname{argmin}}(\Phi(\mathbf{x}^{(\lambda,\ell)})$
    if $\Phi(\mathbf{x}^{(\lambda,\ell)}) = \Phi^*$ **then**
        solution has been reached, stop algorithm
while termination condition(s) not reached **do**
    for each $\mathbf{x}^{(\lambda,\ell)}, \ell = 1, \dots N$ **do**
        update particle's velocity $\zeta^{(\lambda+1,\ell)}$
        update particle's position $\mathbf{x}^{(\lambda+1,\ell)}$
        evaluate updated particle $\Phi(\mathbf{x}^{(\lambda+1,\ell)})$
        if $\Phi(\mathbf{x}^{\lambda+1,\ell)}) = \Phi^*$ **then**
            solution has been reached, stop algorithm
        else if $\Phi(\mathbf{x}^{(\lambda+1,\ell)}) \leq \Phi(\mathbf{x}^{(\lambda,\ell)})$ **then**
            update personal best $\mathbf{x}^{b,(\lambda,\ell)} = \mathbf{x}^{(\lambda+1,\ell)}$
    for each $\mathbf{x}^{(\lambda,\ell)}, \ell = 1, \dots N$ **do**
        if $\Phi(\mathbf{x}^{(\lambda+1,\ell)}) \leq \Phi(\mathbf{x}^{g,(\lambda)})$ **then**
            update global best $\mathbf{x}^{g,(\lambda+1)} = \mathbf{x}^{(\lambda+1,\ell)}$
        else $\mathbf{x}^{g,(\lambda+1)} = \mathbf{x}^{g,(\lambda)}$
    iteration = $\lambda + 1$

The most common implementation of the GPSO algorithm also includes an inertia weight ω which prevents the particles converging to a local optimum and helps to explore a larger search space. The revised velocity update is

$$\zeta_d^{(\lambda+1),\ell} = \omega\zeta_d^{(\lambda,\ell)} + c_1 r_1 (x_d^{b,(\lambda,\ell)} - x_d^{(\lambda,\ell)}) + c_2 r_2 (x_d^{g,(\lambda,\ell)} - x_d^{(\lambda,\ell)}). \tag{3.29}$$

Figure 3.3 illustrates how the velocity and position of a particle is updated in a GPSO. The search space is explored and exploited according to a knowledge of a particle's best position and the overall best achieved by the swarm. The updated particle is then evaluated as a potential solution to a problem using the objective function Φ. The curves in both spaces were drawn at random and do not represent any specific problem.



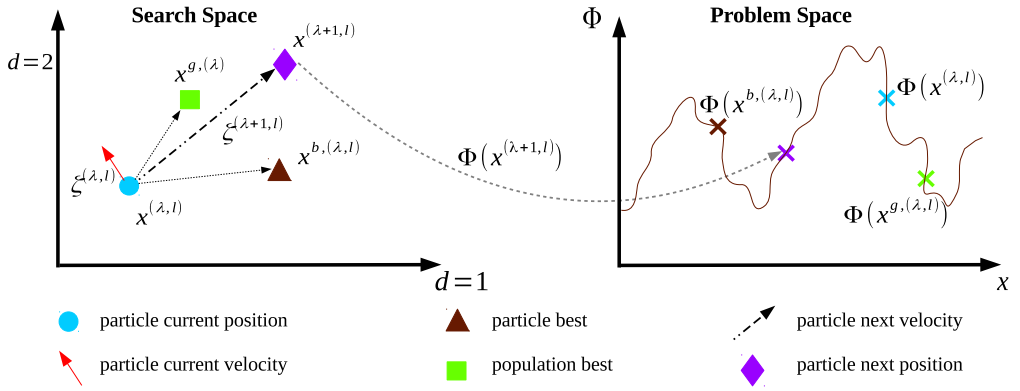
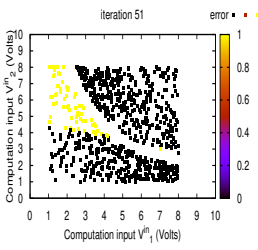


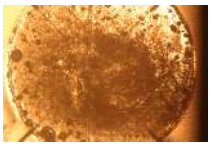
FIGURE 3.3: Update of a particle belonging to the PSO according to its past best and the overall best solution obtained within the swarm.

Research has shown that variations in the constants' values influence the efficiency of the algorithm [59]. In their original paper, Eberhart and Kennedy recommend that $c_1 = c_2 = 2$ [55] and the weight $\omega = 0.76$. In subsequent implementations, $c_1 \neq c_2 = 2$, or in some cases the constants become variables, reducing over time. This type of implementation is introduced when the aim is for the swarm to first explore the search space, and subsequently exploit solutions found [60]. Indeed, small values of c_1, c_2, and ω result in a smaller velocity, and thus a position update within the vicinity of the previous solution. In other cases, the aim can be to increase the influence of the personal, local or global best, with the constants modified accordingly.

The PSO can also vary in terms of how information is shared across the swarm. An alternative to GPSO is the local PSO (LPSO) [61], for which the best position value is shared either through social or geographical neighbourhood [62]. In [63], LPSO shows a faster and better convergence towards a solution to a majority of problems. It is also one of the few adaptation of the PSO that remains very simple. In the same paper, three other PSO variants were tested, all based on adaptive PSO (APSO). Whilst they demonstrate better results than non-PSO algorithms with which they are compared, they are not as effective as GPSO or LPSO. The main issue with most PSO variants is that they exhibit little improvement compared to the simple PSO, but often lose their simplicity [64].

Following this discussion, it was decided to implement only GPSO, in the investigations presented in this work, as for DE in Sec. 3.4.2. The GPSO code used in experiments, implemented with the parameters suggested in [65] was tested against three benchmark optimisation functions: Rosenbrock, Rastrigin and Ackley, before being used in EiM experiments. The results are presented in Appendix B, and were comparable to





those obtained in [65]. This shows that the GPSO used in experiments compares with other GPSOs found in literature. However, given that the implementation parameters used to solve the three test functions have been optimised for these functions specifically, it does not mean that they are optimal for the EiM problem. Different values for the population (swarm) size, inertia weight and constant coefficients were used when PSO was used for *in materio* optimisation. These values were chosen based first on those reported in [65], and subsequently modified based on preliminary results obtained with the materials. It is therefore possible that if the implementation used for the EiM problem was to be tested against benchmark optimisation test functions, results would not be as comparable.

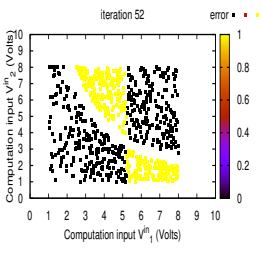
3.5 Hardware Implementation

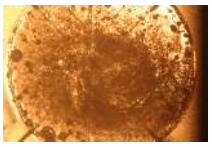
3.5.1 General Characteristics

Two distinct pieces of hardware are currently necessary to conduct EiM experiments: a computer and a hardware interface. The computer is used to run the algorithms described in Section 3.4. The main purpose of the hardware interface is to translate signals produced by the algorithms such that they are able to manipulate the material.

It can be left to either the computer or the hardware interface to interpret output from the material under evolution. The mapping can be either analogue or digital depending on the implementation [66]. The type of input/output signals sent are also implementation dependent. They will differ according to the material used. In experiments where the electrical properties of the material are explored and exploited by the algorithms, as is the case for SWCNT-based composites and the other materials described in Chapter 2, the hardware interface translates the signals sent by the computer into electrical signals.

The final element of the set up is the material itself, which can be seen either as distinct from the hardware interface, or part of it. It must be noted that the hardware interface and computer can have an impact on the solution produced in an experiment as they are included in the optimisation loop. The combination of hardware and material is thus generally considered a black box, but it is important that the properties of the interface be known, in order to make sure that the material is being used and not just the noise produced by the hardware. Other devices such as the microscope light used in





part of the investigations might also have an impact on the material, thus affecting the algorithm's search.

3.5.2 Hardware Interface

Since the field programmable matter array (FPMA) reported in [15, 28], three other hardware interfaces, also referred to as evolvable motherboard (EM) have been constructed. A motherboard must enable the 'translation' of signals from the computer into signals that can be applied to the material. The platform also needs to be designed to obtain a sufficiently large signal-to-noise ratio, such that potentially good solutions are not missed, or noise mis-interpreted as a material output.

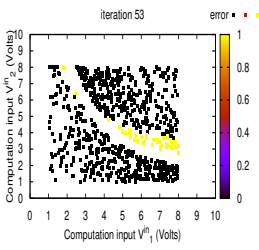
It has also been identified as important for EMs to provide a degree of flexibility in terms of the number, type and level of signal that can be sent to and received from the material [32, 66]. This flexibility increases the number of variables that can be used to manipulate the material.

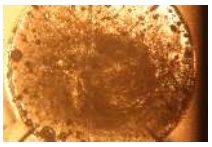
Considering that different materials respond differently to different stimuli, custom-built hardware can be necessary for the purpose of providing the best platform of interaction with a given media and exploiting its properties. Examples of custom-built EMs for biological media are described in [67, 68] and often involve imaging as a method of measuring the material's state under specific stimuli.

Within the context of the Nascence project [31], a versatile EM called mecobo was designed and realised in printed circuit board (PCB), with the aim of enabling exchange of information between any computer and any material which properties can be controlled using electrical signals. Mecobo has been used to investigate which computational problems can be solved through EiM [69], what types of signals should be used in EiM [66], or to compare algorithms' performance [18].

3.5.3 Custom-Built Evolvable Motherboard

The motherboard used to conduct experiments is an updated version of that proposed in [70]. It was originally designed and produced by Dr. M.K. Massey [71] to test materials before they would be sent to other groups from the Nascence project. However, it provided sufficient speed and accuracy to explore the various materials, algorithms and





computational problems used here. In addition, the simple circuit design and components made it easy to repair and improve throughout the course of the investigations, which is why it was chosen over the Mecobo board. The photograph of the EM is presented in Figure 3.4.

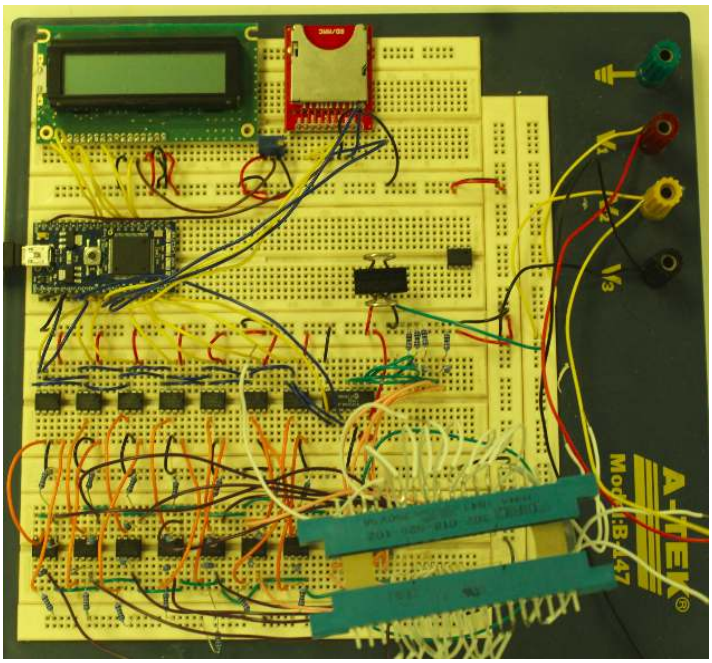
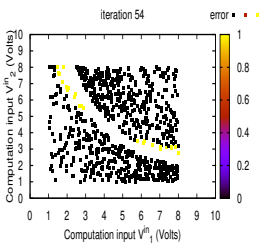


FIGURE 3.4: Photograph of evolable motherboard realised on a breadboard.

The circuit's main components are an *mbed* microcontroller, an SD card and a set of digital-to-analogue converters (DAC). Signals sent from the computer correspond to different variables used to configure the material into a computing device. These include voltage levels and information about where they should be applied [40, 44]. The signals are translated by the *mbed* and voltages are sent to specific locations on the material sample via DACs. The DACs are connected to the material depending upon the type of electrode array used (see Chapter 2).

Input signals are analogue, and direct currents are collected from the material's outputs. Constraints on the variables used to configure the sample are due to the limited flexibility of the EM. In order to reduce noise in the breadboard implementation of the hardware interface (Fig.3.4) and make the whole circuit more resistant to movement and shock, it was later realised in a printed circuit board (PCB), following the design and specifications found in Appendix C. The resulting EM is presented in Figure 3.5.



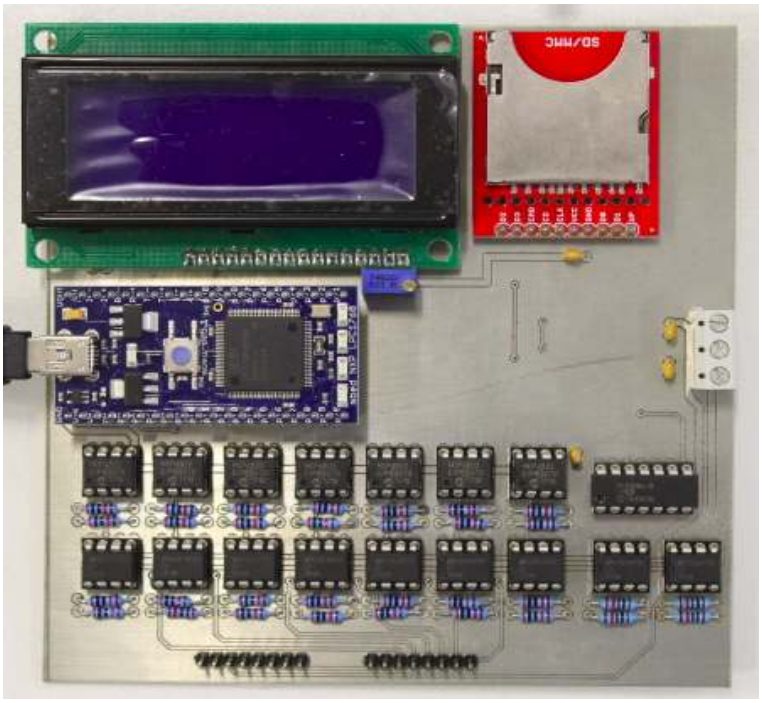


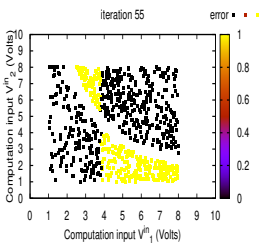
FIGURE 3.5: Photograph of evolvable motherboard realised in PCB.

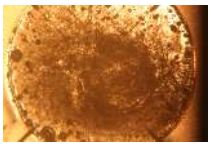
3.6 Experimental Implementation Summary

This chapter has reported the problem formulation used to perform intrinsic, rather than extrinsic, evolution of the materials described in Chapter 2. These materials are treated as black boxes in the formulation. The modification of the input/output relationship of a given material to achieve a specific state which favours the solving of the computational problem was formulated as an optimisation problem. Since no model of the material was used it was not possible to solve the optimisation problems analytically or use an explicit algorithm. Instead, a supervised learning approach was used to find solutions to the optimisation problem, and derivative-free algorithms were used. More specifically, two derivative-free, population-based, stochastic algorithms were: differential evolution (DE) and particle swarm optimisation (PSO). The problem formulation and hardware implementation were designed to allow these algorithms to control configuration signals applied to the material, with the aim of finding a solution to the optimisation problem. Here, both DE and PSO have been referred to as evolutionary algorithms (EAs).

Experimentally, the supervised learning approach detailed in this chapter was implemented using the following steps:

1. Split data defining a computational problem into a **training** and a **verification** set.





2. Initialise the EA-controlled set of decision variables (inc. configuration inputs)[a].

3. Training (repeated until termination criterion reached)

   - apply simultaneously *computation* (training) and *configuration* inputs to the material via the electrode array's terminals[b].
   - measure resulting current across the material[c].
   - translate current outputs into a training error.
   - transfer the error to the computer for evaluation by the EA.
     - *if* termination criterion reached → terminate training.
     - *if* termination criterion not reached → update decision variables.

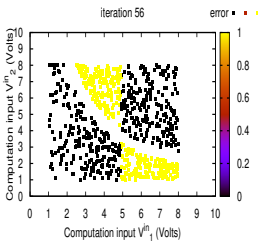
4. No voltages are applied to, or currents measured across, the material for 5 minutes[d].

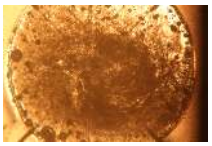
5. Solution Verification (repeated 10 times)

   - apply simultaneously computation (verification set) and optimum configuration inputs to the material.
   - measure resulting current across the material.
   - translate outputs into a verification error.

[a] applying training or verification computation data to the material before training should not result in a current response which minimises the objective function, i.e. the untrained material is, in principles, unable to solve the computational problem at hand.
[b] both configuration and computation inputs are converted into analogue DC voltages using a set of digital-to-analogue converters (DACs) before being applied to the material.
[c] for each training data instance and configuration input applied to the material, the current output is measured three times and the average is used to produce the error. This number balances time per experiment with the effect of potential measurement noise.
[d] a sense of the solution stability is loosely given by this waiting time.

It must be noted that the optimum configuration voltages are part of the optimum solution produced by the algorithm during training, ie: where the minimum error was achieved. Depending on the material used, two types of solution are possible. If it consists in an optimum set of signals producing a material state favouring the solving of the computational problem, it is said that the material has been *optimised*. On the other





hand, if training has modified the morphology of the material, producing structures that, in combination with an optimal set of signals, represent the solution, the material has been *evolved* [72]. Whether optimisation or evolution was performed by the algorithm, it can be said that the material was trained by having its state changed for solving a particular problem, rather than by being able to execute a number of discrete algorithmic steps.

At the start of this chapter, a very general overview of the implementation was presented (fig. 3.1), including the three main hardware components used in EiM: a computer, an evolvable motherboard (EM) and a material. Figure 3.6 presents a more detailed version of the EiM implementation, taking into account the notation proposed for the problem formulation, algorithms and custom-built EM discussed in this chapter. The figure also illustrates one execution of step 3 described above, i.e. one iteration of material training.

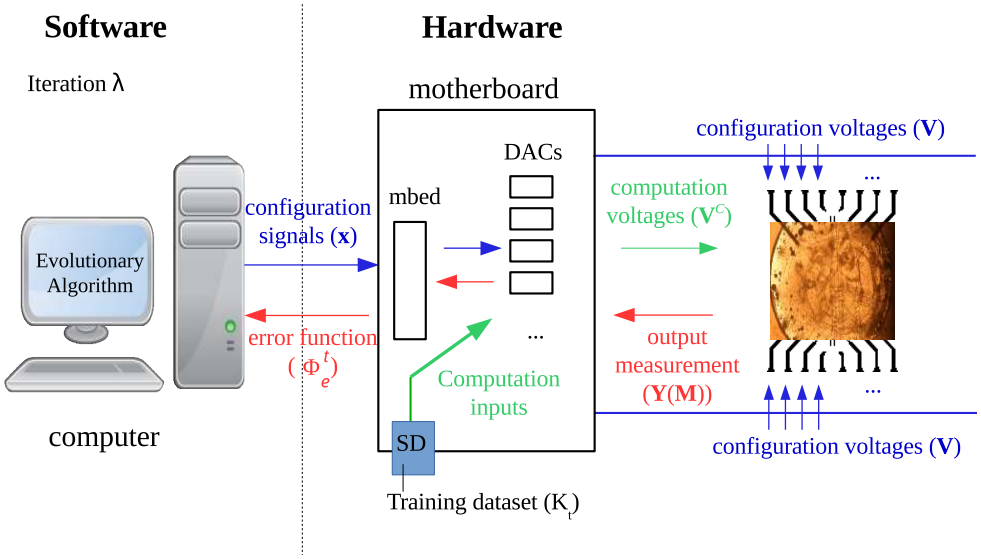
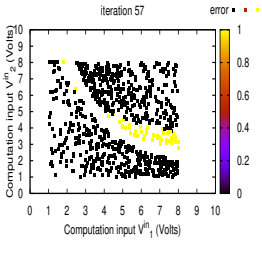


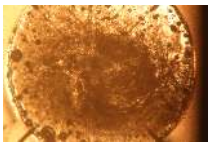
FIGURE 3.6: Implementation of EiM using custom-build hardware and computer

Bibliography

[1] E. Gale, "Single memristor logic gates: From not to a full adder," *arXiv preprint arXiv:1510.05705*, 2015.

[2] F. Alibart, E. Zamanidoost, and D. B. Strukov, "Pattern classification by memristive crossbar circuits using ex situ and in situ training," *Nature communications*, vol. 4, p. 2072, 2013.





[3] A. Serb, A. Khiat, and T. Prodromakis, "Seamlessly fused digital-analogue reconfigurable computing using memristors," *Nature Communications*, vol. 9, no. 1, p. 2170, 2018.

[4] L. Xu, C. Li, and L. Chen, "Analog memristor based neuromorphic crossbar circuit for image recognition," in *Intelligent Control and Information Processing (ICICIP), 2015 Sixth International Conference on*, pp. 155–160, 2015.

[5] K. Greff, R. van Damme, J. Koutnik, H. Broersma, J. Mikhal, C. Lawrence, W. van der Wiel, and J. Schmidhuber, "Unconventional computing using evolution-in-nanomaterio: neural networks meet nanoparticle networks," in *Eighth International Conference on Future Computational Technologies and Applications, FUTURE COMPUTING 2016*, pp. 15–20, 2016.

[6] S. Bose, C. P. Lawrence, Z. Liu, K. Makarenko, R. M. van Damme, H. J. Broersma, and W. G. van der Wiel, "Evolution of a designless nanoparticle network into reconfigurable boolean logic," *Nature nanotechnology*, vol. 10, no. 12, p. 1048, 2015.

[7] M. A. Lones, L. A. Fuente, A. P. Turner, L. S. Caves, S. Stepney, S. L. Smith, and A. M. Tyrrell, "Artificial biochemical networks: Evolving dynamical systems to control dynamical systems," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 2, pp. 145–166, 2014.

[8] J. Friedman, T. Hastie, and R. Tibshirani, *The elements of statistical learning*, vol. 1. Springer series in statistics Springer, Berlin, 2001.

[9] M. Kevin, *Machine learning: a probabilistic perspective*. The MIT press, 2012.

[10] D. Barber, *Bayesian reasoning and machine learning*. Cambridge University Press, 2012.

[11] K. P. Bennett and E. Parrado-Hernández, "The interplay of optimization and machine learning research," *Journal of Machine Learning Research*, vol. 7, no. Jul, pp. 1265–1281, 2006.

[12] A. Defazio, *New Optimisation Methods for Machine Learning (PhD Thesis)*. Australian National University, ArXiv repository, 2014.

[13] NASCENCE project (ICT 317662), "Report on suitable computational tasks of various difficulties," 2013. Deliverable D4.2.

[14] S. Harding, *Evolution in Materio (PhD Thesis)*. The University of York, 2006.

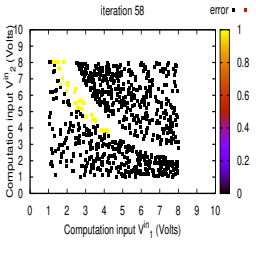
[15] S. L. Harding and J. F. Miller, "Evolution in materio," *Encyclopedia of complexity and systems science*, 2009.

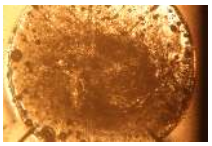
[16] J. F. Miller, S. L. Harding, and G. Tufte, "Evolution-in-materio: evolving computation in materials," *Evolutionary Intelligence*, vol. 7, no. 1, pp. 49–67, 2014.

[17] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.

[18] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[19] M. Dale, J. F. Miller, and S. Stepney, "Reservoir computing as a model for in-materio computing," in *Advances in Unconventional Computing*, pp. 533–571, Springer, 2017.





[20] I. H. Witten and B. A. MacDonald, "Using concept learning for knowledge acquisition," *International Journal of Man-Machine Studies*, vol. 29, no. 2, pp. 171–196, 1988.

[21] R. A. Fisher, "The use of multpitle measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[22] M. Lichman, "UCI machine learning repository," 2013.

[23] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[24] D. Volpati, M. K. Massey, D. Johnson, A. Kotsialos, F. Qaiser, C. Pearson, K. Coleman, G. Tiburzi, D. A. Zeze, and M. C. Petty, "Exploring the alignment of carbon nanotubes dispersed in a liquid crystal matrix using coplanar electrodes," *Journal of Applied Physics*, vol. 117, no. 12, p. 125303, 2015.

[25] M. K. Massey, A. Kotsialos, D. Volpati, E. Vissol-Gaudin, C. Pearson, L. Bowen, B. Obara, D. A. Zeze, C. Groves, and M. C. Petty, "Evolution of electronic circuits using carbon nanotube composites," *Scientific reports*, vol. 6, 2016.

[26] M. Massey, D. Volpati, F. Qaiser, A. Kotsialos, C. Pearson, D. Zeze, and M. Petty, "Alignment of liquid crystal/carbon nanotube dispersions for application in unconventional computing," in *AIP Conference Proceedings*, vol. 1648, p. 280009, AIP Publishing, 2015.

[27] S. Nichele, D. Laketic, and G. Tufte, "Is there chaos in blobs of carbon nanotubes used to perform computation?," *7th International Conference on Future Comp. Tech. and Applications, IN PRESS*, 2015.

[28] S. L. Harding and J. F. Miller, "Evolution in materio: A tone discriminator in liquid crystal," *Congress on Evolutionary Computation, 2004. CEC2004.*, vol. 2, pp. 1800–1807, 2004.

[29] S. L. Harding and J. F. Miller, "Evolution in materio: Evolving logic gates in liquid crystal," *Proc. Eur. Conf. Artif. Life (ECAL 2005), Workshop on Unconventional Computing: From cellular automata to wetware*, pp. 133–149, 2005.

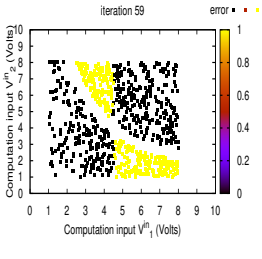
[30] S. L. Harding and J. F. Miller, "Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal," *Evolvable Systems: From Biology to Hardware*, pp. 155–164, 2005.

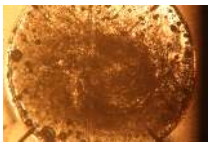
[31] O. R. Lykkebø, S. Harding, G. Tufte, and J. F. Miller, *Mecobo: A Hardware and Software Platform for In Materio Evolution*, pp. 267–279. Cham: Springer International Publishing, 2014.

[32] O. R. Lykkebø and G. Tufte, "Comparison and evaluation of signal representations for a carbon nanotube computational device," in *2014 IEEE International Conference on Evolvable Systems*, pp. 54–60, 2014.

[33] M. Mohid and J. Miller, "Solving even parity problems using carbon nanotubes," in *Computational Intelligence (UKCI), 15th UK Workshop on. IEEE Press*, 2015.

[34] O. R. Lykkebø and G. Tufte, "Evolution-in-materio of a dynamical system with dynamical structures," in *Artificial Life Conference*, p. 242, 2016.





[35] S. S. Farstad, S. Nichele, and G. Tufte, "Towards standalone in-materio devices: Stable logic gates and elementary cellular automata in carbon nanotubes material," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 5246–5254, IEEE, 2016.

[36] S. Nichele, S. S. Farstad, and G. Tufte, "Universality of evolved cellular automata in-materio.," *International Journal of Unconventional Computing*, vol. 13, no. 1, 2017.

[37] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebo, M. K. Massey, and M. C. Petty, "Evolution-in-materio: A frequency classifier using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 46–53, IEEE, 2014.

[38] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving function optimization problems using materials," in *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pp. 1–8, IEEE, 2014.

[39] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving bin packing problems using materials," *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 38–45, 2014.

[40] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[41] F. Qaiser, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, and M. C. Petty, "Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation," in *IEEE Congress on Evolutionary Computation (CEC)*, pp. 5255–5261, IEEE, 2016.

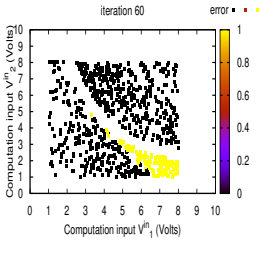
[42] NASCENCE project (ICT 317662), "Advanced evolutionary algorithms," 2014. Deliverable D3.3.

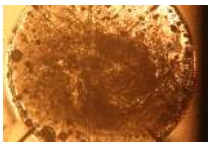
[43] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes.," *International Journal of Unconventional Computing*, vol. 10, no. 5-6, pp. 473–497, 2014.

[44] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Training a carbon-nanotube/liquid crystal data classifier using evolutionary algorithms," *Unconventional Computation and Natural Computation Conference: 15th International Conference, UCNC 2016*, pp. 130–141, 2016.

[45] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Data classification using carbon-nanotubes and evolutionary algorithms," *International Conference on Parallel Problem Solving from Nature*, pp. 644–654, 2016.

[46] E. Ozcan and C. Mohan, "Analysis of a simple particle swarm optimization system," *Proc. Conf. Artificial Neural Networks in Engineering*, no. 1998, pp. 253–258, 1998.





[47] S. Talukder, "Mathematical Modelling and Applications of Particle Swarm Optimization," *Mathematical Modelling*, no. February, 2011.

[48] V. Plagianakos, D. Tasoulis, and M. Vrahatis, "A review of major application areas of differential evolution," in *Advances in differential evolution*, pp. 197–238, Springer, 2008.

[49] R. Poli, "Analysis of the publications on the applications of particle swarm optimisation," *Journal of Artificial Evolution and Applications*, vol. 2008, 2008.

[50] J. Vesterstrom and R. Thomsen, "A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems," in *Proceedings of the 2004 Congress on Evolutionary Computation (IEEE Cat. No. 04TH8753)*, vol. 2, pp. 1980–1987, IEEE, 2004.

[51] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[52] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[53] S. Das and P. N. Suganthan, "Differential evolution: A survey of the state-of-the-art," *IEEE transactions on evolutionary computation*, vol. 15, no. 1, pp. 4–31, 2011.

[54] M. E. H. Pedersen, "Good parameters for differential evolution," tech. rep., Technical report, Hvass Computer Science Laboratories, 2010.

[55] R. C. Eberhart, J. Kennedy, *et al.*, "A new optimizer using particle swarm theory," *Proceedings of the sixth international symposium on micro machine and human science*, vol. 1, pp. 39–43, 1995.

[56] R. Eberhart and J. Kennedy, "Particle swarm optimization," in *Proceedings of the IEEE international conference on neural networks*, vol. 4, pp. 1942–1948, Citeseer, 1995.

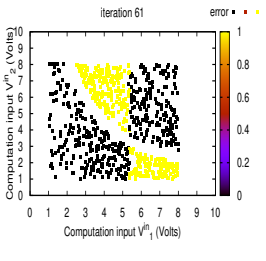
[57] J. Kennedy and R. Eberhart, "Particle swarm optimization," *Neural Networks, 1995. Proceedings., IEEE International Conference on*, vol. 4, pp. 1942–1948 vol.4, 1995.

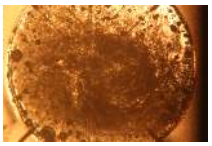
[58] J. Kennedy, "The particle swarm: social adaptation of knowledge," *Proceedings of 1997 IEEE International Conference on Evolutionary Computation (ICEC '97)*, pp. 303–308, 1997.

[59] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," in *Proceedings of the 2000 congress on evolutionary computation. IEEE CEC00*, vol. 1, pp. 84–88, IEEE, 2000.

[60] Y. Tuppadung and W. Kurutach, "Comparing nonlinear inertia weights and constriction factors in particle swarm optimization," *International Journal of Knowledge-Based and Intelligent Engineering Systems*, vol. 15, no. 7, pp. 65–70, 2011.

[61] J. Kennedy and R. Mendes, "Population structure and particle swarm performance," 2002.





[62] J. Chen, Z. Qin, Y. Liu, and J. Lu, "Particle Swarm Optimization with Local Search," in *Neural Networks and Brain, 2005. ICNN B '05. International Conference on*, vol. 1, pp. 481–484, Oct. 2005.

[63] A. Poole and A. Kotsialos, "Swarm intelligence algorithms for macroscopic traffic flow model validation with automatic assignment of fundamental diagrams," *Applied Soft Computing*, vol. 38, pp. 134–150, 2016.

[64] Y. Tan and J. Zhang, "Magnifier Particle Swarm Optimization," in *Nature-Inspired Algorithms for Optimisation* (R. Chiong, ed.), vol. 193 of *Studies in Computational Intelligence*, pp. 279–298, Springer Berlin Heidelberg, 2009.

[65] M. E. H. Pedersen, "Good parameters for particle swarm optimisation," tech. rep., Technical report, Hvass Computer Science Laboratories, 2010.

[66] O. R. Lykkebø, S. Nichele, and G. Tufte, "An investigation of square waves for evolution in carbon nanotubes material," *13th European Conference on Artificial Life*, 2015.

[67] M. Amos, I. M. Axmann, N. Blüthgen, F. de la Cruz, A. Jaramillo, A. Rodriguez-Paton, and F. Simmel, "Bacterial computing with engineered populations," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 373, 2015.

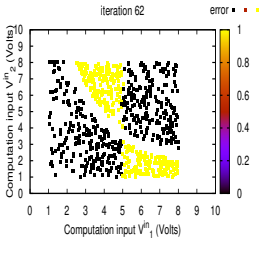
[68] J. Jones, J. Whiting, and A. Adamatzky, "Quantitative transformation for implementation of adder circuits in physical systems," *Biosystems*, vol. 134, pp. 16–23, 2015.

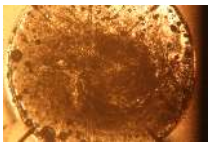
[69] M. Mohid, *Evolution-In-Materio: Solving Computational Problems Using Materials (PhD Thesis)*. University of York, 2015.

[70] NASCENCE project (ICT 317662), "Report on materials systems and electrical behaviour," 2013. Deliverable D1.1.

[71] M. K. Massey, "Internal durham nascence meeting reports." Internal report, unpublished, 2013-2016.

[72] NASCENCE project (ICT 317662), "Report on computational capabilities," 2014. Deliverable D1.2.





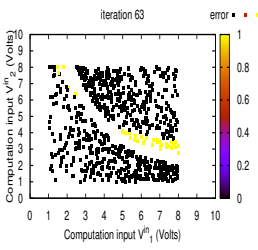
Chapter 4

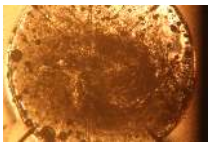
Solving Synthetic Binary Classification Problems with Carbon-Nanotube / Liquid Crystal Composites

4.1 General Overview

The first question this chapters addresses is whether solutions to computation problems can be found through training, or evolution, of the single-walled-carbon-nanotube / liquid crystal (SWCNT/LC) composites described in Chapter 2, using the implementation detailed in Chapter 3. The computation problems consist of five synthetic binary classification problems (BCPs) of increasing complexity.

BCPs have been used in investigations where the capacity of evolution in materio (EiM) and reservoir computing in materio (RCiM) to transform solid SWCNT/polymer composites into linear and non-linear classifiers has been demonstrated. Results obtained with these two frameworks have been good proof-of-concept [1] or comparable with state-of-the-art algorithms [2] or optimal [3]. The differences in results were partly attributed to the framework used and the different BCPs' complexity. However, it was also observed that a number of implementation parameters (SWCNT concentration [4], electrode number, applied waveform,...) had an impact on the training efficiency of SWCNT/polymer classifiers, in terms of training speed and solution accuracy.





The second question addressed in this chapter is therefore concerned with the parameters of the EiM implementation. After proof-of-concept results have been obtained for the SWCNT/LC composite, and if the results are not optimal, the impact of:

- SWCNT concentrations,
- evolutionary algorithms (EAs),
- and problem formulation parameters,

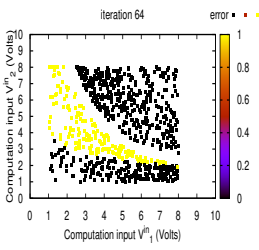
on the training speed and solution accuracy obtained during experiments with SWCNT/LC composites are investigated.

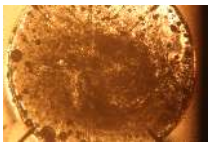
The field of EiM being fairly new, a number of implementation parameters are chosen ansatz or arbitrarily in all EiM-related investigations. It is also the case here. However, the aim of the second set of experiments is to produce empirical justifications for some of the implementation choices and find an optimum combination of concentration/EA/problem formulation for the solving of the BCPs in SWCNT/LC composites.

This chapter is organised in two parts. The four BCPs along with the formulation used to solve them *in materio* are presented in the first part (Sec. 4.2-4.4), along with a description of two methods used for result analysis. The second part is concerned with the presentation and analysis of results obtained in experiments. Control experiments are introduced along with proof-of-concept results. This is followed by a comparison of results between varying SWCNT concentrations, algorithms and problem formulation. Finally a summary of results and analysis concludes this chapter.

4.2 Binary Classification Problems (BCPs)

A BCP is a type of classification problems commonly used to test new machine learning algorithms or computing frameworks. This use is motivated by the relative simplicity of some BCPs compared to multi-class classification problems, combined with the wide range of applications for the fast and accurate solving of BCPs, from medicine [5–7] to meteorology [5, 8, 9]. The same motivations justified the choice of BCPs in the investigations presented here. The possibility to solve BCPs using EiM has already been demonstrated using solid SWCNT/polymer composites [1, 3]. Therefore the potential advantages of evolved SWCNT/LC classifiers over solid SWCNT/polymer classifiers are explored on the one hand, and on the other the impact of the training process on the SWCNT/LC composite.





The BCPs used here were generated by a source, structuring data into two different classes, instead of being obtained from a repository. They are therefore referred to as synthetic as they do not represent real-life problems. The synthetic BCPs belong to a finite dimensional space spanned by the number of distinctive characteristic features measured or observed as the process generates them. The task at hand is to design classifiers that assign each newly generated datum to one of the two possible classes. Measurement ambiguity, complex decision boundaries and subclass structures all contribute to the problem's difficulty [10].

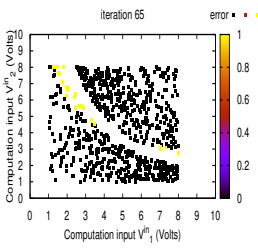
4.2.1 Characteristics of the Synthetic BCPs

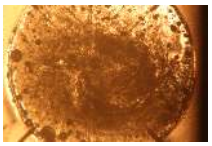
Parameters

A typical training and verification approach is followed for assessing a material's capacity to act as a classifier. Five synthetic BCPs with $n_1 = 2$ attributes were created. In total $K_{tot} = 4800$ points, or instances, were randomly generated. The training sets contain $K_t = 800$ points and the verification sets $K_v = 4000$ points. The total number of data instances and ratio of training to verification instances were chosen arbitrarily, whilst the shape of the class' boundaries were chosen to represent different levels of complexity.

A data point is defined by the pair of coordinates $(V_1^{\mathcal{C}}, V_2^{\mathcal{C}})$ belonging to either class 1, in which case $\mathcal{C}\left[(V_1^{\mathcal{C}}, V_2^{\mathcal{C}})\right] = 1$, or class 2, in which case $\mathcal{C}\left[(V_1^{\mathcal{C}}, V_2^{\mathcal{C}})\right] = 2$. The synthetic problems' classes are illustrated in Figure 4.1. Training and verification data are generated independently and distributed randomly within each class's boundary. The five datasets are called SC, V1C, NLC, NNLC and MC. They are differentiated by the distance between their classes and the shape of the separating boundary.

In the simplest BCP, referred to as the SC problem, the two classes are fully separable and arranged in rectangular-shaped regions defined here. Both training and verification dataset defining SC are illustrated in Fig.4.1(a) and (b) respectively. The other three problems are separable in two dimensions. However, their classes overlap if only one dimension is considered, i.e. $V_1^{\mathcal{C}}$ or $V_2^{\mathcal{C}}$ rather than both. The V1C problem's two triangular-shaped classes are separated by a diagonal boundary, as illustrated in Fig.4.1(c) whilst Fig.4.1(d) shows that the NLC problem's classes are separated by a hyperbolic curve. The classes of the NNLC problem are separated by an S-shaped boundary (Fig.4.1(e)).





In the last BCP, referred to as MC, the classes are partially merged. Fig.4.1(f) illustrates the MC problem training dataset, including the area, containing 6.6% of all data points, where the two classes overlap. Instances distributed within this overlapping area are effectively indistinguishable in either or both dimensions.

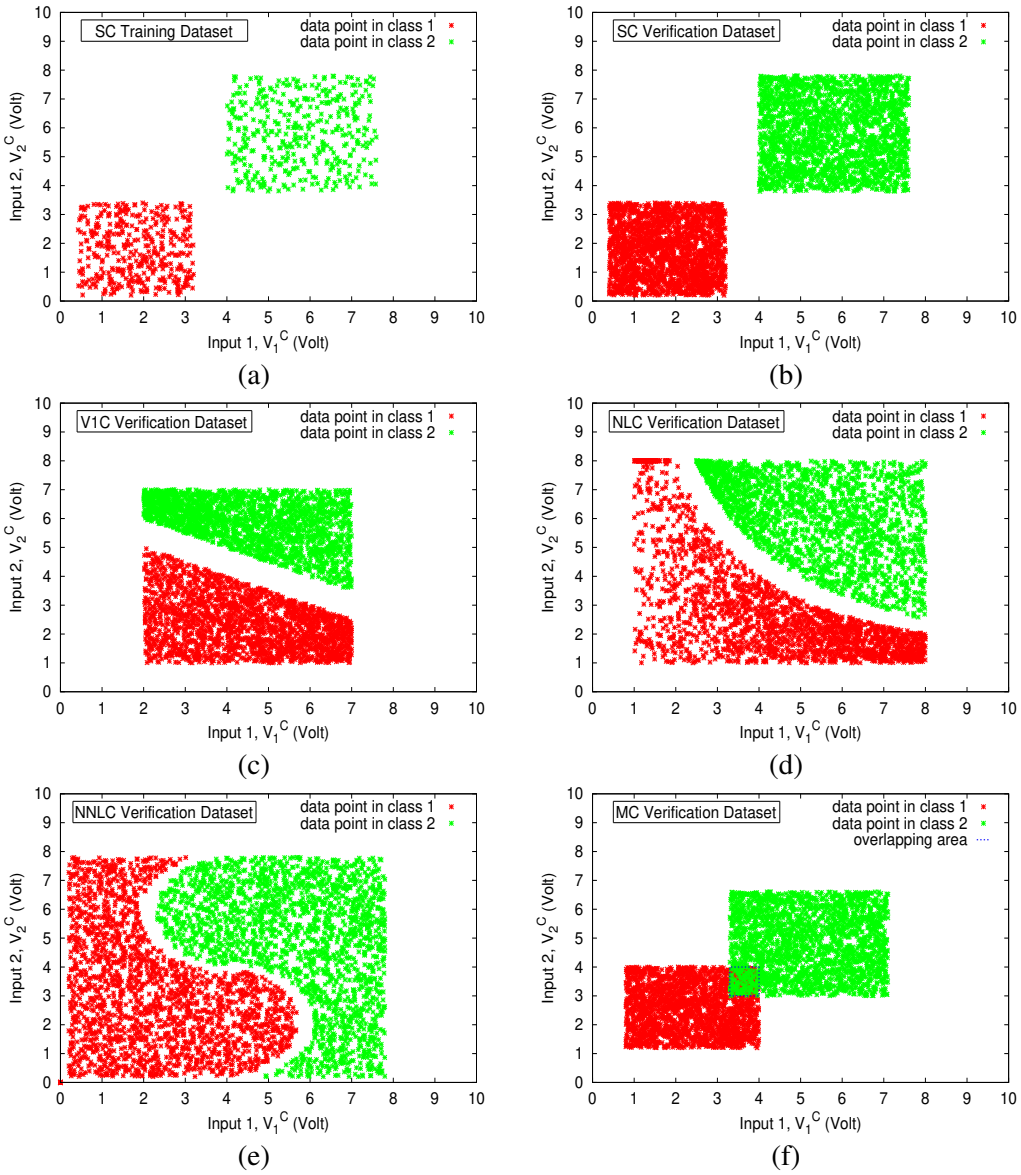
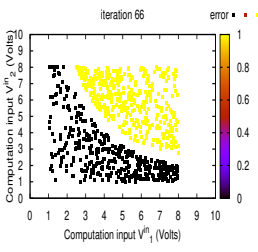
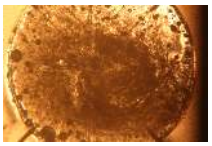


FIGURE 4.1: (a) Training dataset for SC and verification datasets for (b) SC, (c) V1C , (d) NLC, (e) NNLC and (f) MC.

Across the five problems, both training and verification datasets are balanced datasets, in the sense that the total number of instances is split equally between class 1 and class 2. As a result, if instances are assigned randomly to a class the classification error will be around 50%. This is the % of error expected at the start of an experiment, when a material is in its initial state, i.e. in a state that does not favour the classification of data.





A solution resulting in 50% error is therefore the worst possible solution, as it effectively demonstrates no improvement of the classifier from its original untrained state, and the material is effectively performing a random coin toss or assigning all instances to one class. On the other hand, a solution inducing the correct classification of all instances from the separable datasets, thereby resulting in 0% error, is optimum. For all the BCPs of Fig. 4.2, if the error is 100%, a solution to a maximisation rather than a minimisation problem has been found, i.e. the classes have been inverted. However, the resulting classifier has identified the correct separating boundary between classes and the solution can be considered good.

The optimum for the linearly and non-linearly separable datasets is different from the MC dataset's optimum. In the latter case, it is not possible to classify instances contained in the area where classes merge. The minimum error for this problem is therefore one where all instances outside of the overlap are correctly classified and the 6.6% of instances within the overlap are classified at random, resulting in 3.3% error.

Complexity

In terms of eq. (3.1) from Chapter 3, Section 3.2.1, describing the domain definition, $\mathcal{A}$, of the computation inputs,

$$\mathcal{A} = [0,8] \times [0,8] = [0,8]^2 \subset \mathbb{R}^2 \text{ with } \mathcal{D} = \{1,2\}\,, \tag{4.1}$$

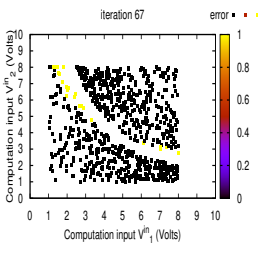
where 0 and 8 define the minimum and maximum. These values were chosen based on hardware limitations, which only allows positive voltages up to 8 Volts to be applied to the material.

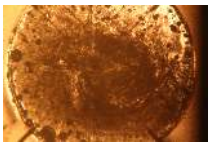
For an arbitrary BCP, a common measure of complexity is the Fisher complexity measure, which evaluates the level of separation between classes in a dataset. For each feature j of a dataset, F_j is the Fisher criterion, [10, 11], defined as

$$F_j = \frac{\left(\mu_{1,j} - \mu_{2,j}\right)^2}{\sigma_{1,j}^2 + \sigma_{2,j}^2} \tag{4.2}$$

where $\mu_{i,j}$ and $\sigma_{i,j}$ are the mean and standard deviation of feature j for class $i = 1$ or 2.

Typically, a problem's Fisher complexity measure is taken as the maximum of all F_j, [10]. However, within the framework of EiM, the material must be trained to be able





to discern the data along all features simultaneously without explicit preference to those features with high F_j. Hence, the Fisher complexity for a BCP is taken as the sum of the feature complexities F_j, i.e.

$$F = \sum_{j=1}^{n_1} F_j = F_1 + F_2, \tag{4.3}$$

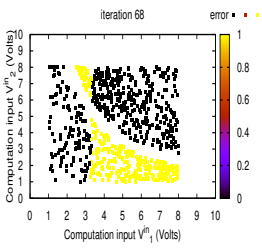
and a high value of F corresponds to a low problem complexity.

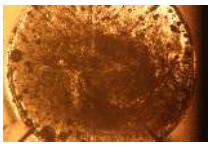
The four problems are reported in ascending order of complexity in Table 4.1, along with the number of features n_1 and the total number of training and verification instances. It must be noted that F is effectively a measure of the classes' separability in terms of distance from and spread around their centre of mass. However, it does not take into account linearity of the boundary separating the classes, which also contributes to the problem's complexity [10]. This explains why the NNLC problem has a higher Fisher criterion than the NLC problem, despite having a more non-linear boundary which makes it more complex, and in theory, more difficult to solve.

TABLE 4.1: Synthetic BCPs and their parameters, arranged in ascending order of complexity.

BCP	n_1	$K_t + K_v$	F_1	F_2	F
SC	2	$4,800$	8.842	7.402	16.244
V1C	2	$4,800$	$2.15E-5$	9.198	9.198
MC	2	$4,800$	3.617	2.862	6.479
NLC	2	$4,800$	0.097	2.220	2.317
NNLC	2	$4,800$	0.342	2.491	2.833

The SC, V1C and MC problems were designed to assess the ability of the EiM framework to evolve the material into different linear classifiers. The NLC and NNLC problems were designed subsequently, as a more complex task, that of evolving the material into a non-linear data classifier. The evaluation of the different classifiers' complexity, however, is based on complexity analysis applied to the conventional computing framework and devices, it might not reflect the complexity of the *in materio* classifiers, i.e. it might be more 'natural' for the SWCNT/LC device to solve non-linear BCPs than linear ones.





4.2.2 Formulation of the EiM Training Problem for the BCP

The BCP EiM training optimisation problem, is based on the general formulation presented in eqs. (3.12)–(3.14) of Chapter 3. When used for the BCP, the interpretation scheme $S_{\mathcal{C}}$ which translates the current measured across the material into a class (eq. (3.7)), takes the form of the following threshold rule:

$$S_{\mathcal{C}}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}(\mathbf{M}), \mathbf{R}\right) = \begin{cases} 1 & \text{if} \quad h\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}(\mathbf{M}), p\right) \leq R_1 \\ 2 & \text{if} \quad h\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}(\mathbf{M}), p\right) > R_1 \end{cases} \tag{4.4}$$

where h is a problem dependent real valued function. The continuous decision variable, R_1 acts as a threshold in $S_{\mathcal{C}}$. p is used to choose the electrode assignment for the input voltages. Both are components of the vector of auxiliary quantities $\mathbf{R}$, which, for EiM problem formulation used to solve the BCP is

$$\mathbf{R} = [R_1\, p]^T . \tag{4.5}$$

The combination of $\mathbf{R}$ and the vector of configuration inputs $\mathbf{V} = [V_1 \ldots V_{10}]^T$ form part of the vector of configuration variables $\mathbf{x}'$ controlled directly by the optimisation algorithms. As previously mentioned, the full vector, $\mathbf{x}$, also includes the material state $\mathbf{M}$ which is indirectly controlled by the algorithm through $\mathbf{R}$ and $\mathbf{V}$.

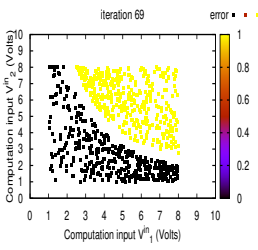
In order for (4.4) to be applied using the evolvable motherboard (EM) described in Chapter 3, the vector of measured responses $\mathbf{Y}(\mathbf{M})$ consists of two direct current measurements $I_1(\mathbf{M})$ and $I_2(\mathbf{M})$ (Amp) taken from two terminal electrodes. The locations of these terminals remain the same and does not change during training, as this is a hard wired feature of the motherboard. Thus,

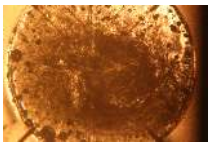
$$\mathbf{Y}(\mathbf{M}) = \left[I_1(\mathbf{M})\ I_2(\mathbf{M})\right]^T . \tag{4.6}$$

In the most basic interpretation scheme used for the BCPs of subsection 4.2.1, the function h in eqn. (4.4) takes the form

$$h^{(1)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\right) = h^{(1)}\left(\mathbf{Y}\right) = \frac{I_1}{I_2} \tag{4.7}$$

where the dependence on $\mathbf{M}$ and p are dropped for the sake of clarity.





Since little is known regarding the impact of the interpretation scheme on a material's ability to solve a computational problem, the simple form of $h^{(1)}$ presented in (4.7) was chosen as a starting point for investigations relating to the interpretation scheme. The aim was to make the dependence on the material as direct as possible through the measured outputs, in order to prevent the algorithms from by-passing the material.

It must be noted that it remains possible for the material to be by-passed. For example, in the case of a hardware failure, the values of I_1 and I_2 could be artificially created by the faulty components and thereby independent of the material state. In order to ensure that this is not the case, preliminary tests were performed before the start of each experiment. A voltage level was applied to the material through all the input electrodes. The current output recorded by the EM was compared to that recorded using a multimeter. If the difference between the two current values was large, the experiment was not carried out until the source of failure had been addressed.

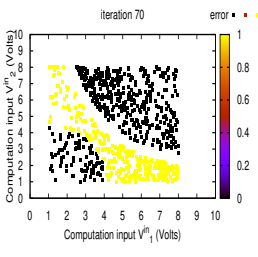
Equations (4.4) and (4.7) allow the definition of the error function (3.10), given in Chapter 3, for an arbitrary computation input $\mathbf{V}^{\mathcal{C}}$ as

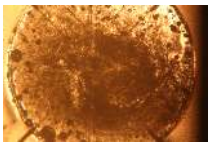
$$\epsilon_{\mathbf{x}}\left(\mathbf{V}^{\mathcal{C}}\right) = \begin{cases} 0 & \text{if} \quad S_{\mathcal{C}}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}, \mathbf{R}\right) = \mathcal{C}\left(\mathbf{V}^{\mathcal{C}}\right) \\ 1 & \text{if} \quad S_{\mathcal{C}}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}, \mathbf{R}\right) \neq \mathcal{C}\left(\mathbf{V}^{\mathcal{C}}\right). \end{cases} \tag{4.8}$$

The objective function evaluation for the optimisation problem (3.12)–(3.14) in Chapter 3 is constructed by averaging the error defined in (4.8) over all points in the training dataset during the SWCNT/LC training. Similarly, (4.8) is used for assessing the solutions' quality against the verification datasets, resulting in the verification error of (3.22).

Preliminary investigations suggested that allowing the EAs to select where to apply the input signals on the material during training enabled them to find better solutions to the BCPs. This is consistent with the discussion reported in [3, 12] for SWCNT/poly(butyl meta-acrylate) (PBMA) devices. The ability to switch the location (electrode terminal of the glass slide) where $\mathbf{V}^{\mathcal{C}}$ and $\mathbf{V}$ are applied was therefore identified as important.

In order to allow algorithms to chose the set of electrodes to which computation and configuration voltages are applied, a continuous decision variable p in (4.4) was introduced. Since the number of all possible assignments of terminals to computation and configuration inputs is very large, the value of the variable p represents only a possible





pair of glass slide terminals where the components of $\mathbf{V}^{\mathcal{C}}$ are to be applied. According to this scheme, there are ${}^{14}P_2 = 182$ possible connection assignments, and p is defined over the interval $[1,\ 182]$. At every function evaluation λ, the variable p is rounded to the nearest integer, corresponding to a unique feasible assignment. The configuration and computation inputs are then applied to the material following this assignment.

An example is illustrated in Fig. 4.2 where an array schematic with sixteen electrodes is used along with three BCPs of two computation inputs each. The two output measurements I_1 and I_2 are collected from the two fixed locations (indicated by a cross) 1 and 9 (clockwise numbering of terminals starting from the bottom left corner). The position of these electrodes is determined before training and cannot be changed by the optimisation search algorithm due to hardware constraints. This leaves fourteen terminals free for the application of the problems' $n_1 = 2$ computation and the $n_2 = 12$ possible configuration inputs.

The most common solution for the three BCPs, in terms of configuration input locations, is shown in Fig. 4.2. In the case of V1C, $V_1^{\mathcal{C}}$ is assigned to electrode 1, i.e. $V_1^{\mathcal{C}} \rightarrow 1$ and $V_2^{\mathcal{C}} \rightarrow 2$ this directly results to the following assignment for the configuration inputs (indicated by black bullets): $V_1 \rightarrow 3$, $V_2 \rightarrow 4$, $V_3 \rightarrow 5$ $V_4 \rightarrow 6$, $V_5 \rightarrow 7$, $V_6 \rightarrow 10$, $V_7 \rightarrow 11$, $V_8 \rightarrow 12$, $V_9 \rightarrow 13$, $V_{10} \rightarrow 14$, $V_{11} \rightarrow 15$ and $V_{12} \rightarrow 16$. In practice not all terminals are used at all time, either because it is not necessary or because it is not possible (e.g: manufacturing faults).

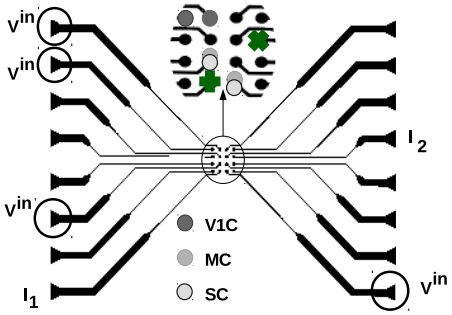
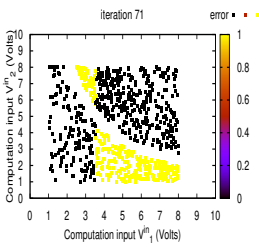
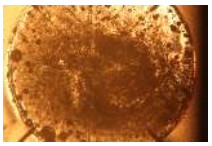


FIGURE 4.2: Example of computation inputs and outputs assignments for various BCPs.





4.3 Statistical Tools for Results Evaluation

In order to study the efficiency of the EiM training, a number of W experiments are conducted for each set of investigations which, except if otherwise stated, start from different and initially untrained material samples. In each experiment $j = 1, \ldots, W$, both training and verification are performed yielding the best training error $\Phi_{e,j}^{t,*}$ at iteration λ_j^*, and the corresponding verification error $\Phi_{e,j}^{v}$, obtained after training has ended. These errors are calculated according to eq.(3.22) from Chapter 3

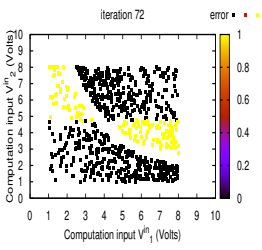
Based on these experimental results, the following efficiency metrics are used

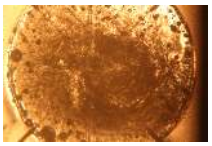
$$\begin{aligned} &\Phi_e^{t,*} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W} \Phi_{e,j}^{t,*}, \quad \Phi_e^{v,*} = \min_{j=1,\ldots,W} \Phi_{e,j}^{v} \\ &\overline{\Phi_e^v} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W} \Phi_{e,j}^{v} \qquad \text{and } \sigma_{\Phi_e^v}, \end{aligned} \tag{4.9}$$

$\overline{\Phi_e^v}$ in conjunction with the standard deviation $\sigma_{\Phi_e^v}$ provide a measure of the results' reproducibility on different material samples. Added to the best iteration averaged over experiments $\lambda_e^* = \frac{1}{W}\sum_{j=1}^{W} \lambda_{e,j}^*$, these metrics provide information regarding the speed, accuracy and reproducibility of the training process. EiM training is considered efficient if the errors averaged across experiments are close to the problem's optimum (0% for all separable classes, 3.3% for the merged classes problem) and the standard deviation in verification error is zero. In addition, an efficient implementation is expected to achieve the lowest possible error and standard deviation as fast as possible, i.e. with the smallest possible λ_e^*. It must be noted, however, that a $\lambda_e^* = 0$ would not be a good indicator of EiM training efficiency, as it suggests that the material is in a state favouring good (= accurate) classification of data pre-training, or alternatively, it might suggest issues with the hardware or software.

Another notion of EiM training efficiency is given by how well a solution generalises, i.e. how low is the difference between the optimum training and verification errors. For each experiment, $|\Phi_{e,j}^{t,*} - \Phi_{e,j}^{v,*}|$ is the absolute difference between the best training error obtained in the experiment and the best verification error obtained accross the verification tests. The average of this difference over the W experiments becomes

$$\overline{d_\Phi} = \frac{1}{W}\sum_{j=1}^{W} |\Phi_{e,j}^{t,*} - \Phi_{e,j}^{v,*}|, \tag{4.10}$$



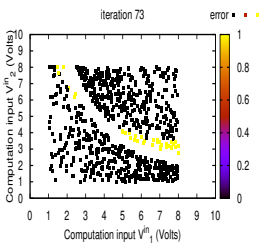


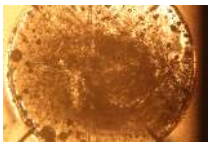
and if $\overline{d_\Phi} = 0$, the solution obtained during training has generalised perfectly to unseen data over all W experiments. It must be noted that for each EiM experiment, the verification tests from which $\Phi^{v,*}_{e,j}$ is taken, and $\Phi^{v}_{e,j}$ computed, are performed post-training. They ideally consist in testing the best solution evolved during training $\mathbf{x}^*$ against instances from the verification dataset which are different from those of the training dataset. In practice, the partial solution $\mathbf{x}^{'*} = [\mathbf{V}p\mathbf{R}]^T$ is applied to a material which may or may not be in the best state $\mathbf{M}^*$. The verification tests are repeated ten times. Further discussion relating to the verification tests were reported in Chapter 3.

Between $W = 5$ and $W = 20$ experiments were undertaken, depending on the investigation. The number of experiments was limited by the time required to perform each one of them. For example, training one sample of material to solve a problem, with $K_t = 800$, for two hundred iterations ($\Lambda = 200$) and an algorithm with a population of ten individuals (i.e. ten potential solutions to the optimisation problem to be tested), took 3 hours and 30 minutes, in addition to the time required to prepare the sample.

When a relatively small number of experiments is undertaken, comparing means, or even medians, can be misleading, especially if the set of results contains outliers. For example, two sets of results might have very different means over three experiments, but over twenty experiments, this difference becomes negligible. Where possible, comparisons between results obtained with different experimental implementations of EiM were therefore complemented by statistical significance tests, in order to give a reasonable indication of the statistical significance that can be allocated to the comparison, i.e. whether the differences observed are representative or due to under sampling.

The two-tailed Mann-Whitney U-test, a non-parametric statistical significance test, was used in result analysis [13], due to the small sample size, and the fact that result data did not follow a normal distribution. In order to implement this test, the sets of results compared must be independent. The test verifies whether two samples of results belong to the same distribution, i.e. if the samples' mean and standard deviation would be the same were the number of results infinite. A p-value < 0.05 indicates that the differences between two sets of results are statistically significant, whilst they are not if p ≥ 0.05. This is consistent with other work in the field of EiM [14, 15].





4.4 Evaluating the Rate of Change in Material Morphology

The SWCNT/LC composite morphology varies under an applied electric field. This is reported in [16] where it is observed that the bundling and alignment of SWCNTs within the LC matrix can be linked to electrical characteristics of the material. In addition, important differences in rate of change in material morphology at different parts of the training process were reported in [17]. These observations suggest that changes in material morphology induced by EA-controlled voltages applied during training have an impact on the capacity of the composite to be evolved into a computing device. In order to provide more information regarding this impact, a measure of change in material morphology is introduced here. The aim is to quantify this phenomenon, allowing it to be compared to the quantitative data assessing the material's computational capabilities.

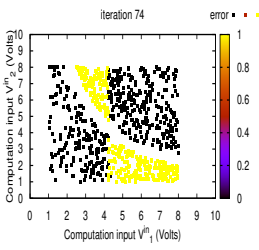
The mean-squared error (MSE) between images, from iteration to iteration, would be one way to estimate the change in material morphology. The MSE between two images is a measure of the difference in the intensity of each pixel in one image with the intensity of the corresponding pixel in the other image. This can reflect changes in luminosity, contrast or structure. However, using the MSE, large differences in contrast or luminosity between two images will result in a large mean-squared difference, even if there was no changes in the images' structure. Thus, changes in lighting, or unexpecting blurring of the microscope lens during an experiment would affect the results.

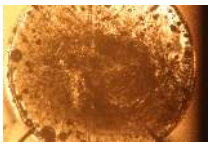
Instead, the structural similarity index measure (SSIM), developed in [18] was used here. This system compares images over three metrics: luminosity, contrast and structure. The SSIM is formulated as:

$$\mathrm{SSIM}(\pi_1, \pi_2) = \frac{\left(2\mu_{\pi_1}\mu_{\pi_2} + c_1\right) - \left(2\sigma_{\pi_1\pi_2} + c_2\right)}{\left(2\mu_{\pi_1}^2 + \mu_{\pi_2}^2 + c_1\right)\left(2\sigma_{\pi_1}^2 + \sigma_{\pi_2}^2 + c_2\right)} \tag{4.11}$$

where π_1 and π_2 correspond to two non-negative signals from two images Π_1 and Π_2 which are being compared. These signals are generally pixel data from the same portion of each image. The two constants c_1 and c_2 are used to avoid instability that might arise due to little luminosity or contrast within the image portions analysed. The SSIM is defined between -1 and 1. A value of 0 indicates no structural similarity between two images, whilst a SSIM=1 or -1 indicates two identical images.

In theory, comparisons of the three metrics can be obtained separately. However,





since changes in luminosity and contrast can be due to changes in the LC's orientation and not just to variations in the environment, equation (4.11) allocates the same importance to all. Further details regarding this formulation are presented in [18].

During experiments, one photograph was taken half-way through each iteration. The SSIM between each photograph thus reflects the impact of ten sets of configuration voltages applied sequentially on the material, rather than the difference between two solutions. Over the whole training process, the changes in SSIM value form a rate of change (RoC) in material morphology which can be compared to the convergence in the objective function, the configuration voltage trajectories or the changes in current outputs collected across the material.

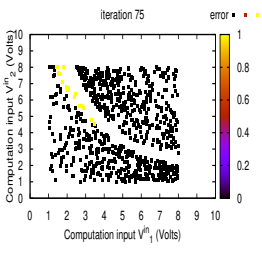
4.5 Control Experiments

4.5.1 Motivation

The control experiments were undertaken with the aim of providing a basis of comparison for all subsequent experiments. The potential bias produced by the EM's components or the micro-electrode array was investigated using a bare array as 'evolvable' material. In this case, the algorithms evolve an open circuit. LC-only samples, drop-cast on the micro-electrode array, were used to determine whether the SWCNT/LC composites' ability to solve computational problems was dependent on the presence of SWCNTs. Finally, the benefit gained, in terms of classification accuracy, from using SWCNT-based samples over more conventional components was evaluated. In this case, the usual micro-electrode array and material were replaced by a linear resistor array. All results were compared with 0.05 wt % SWCNT/LC samples used as proof-of-concept for the liquid composite.

4.5.2 Experimental Implementation

The characteristics of the three materials and the experimental process followed are detailed in Chapter 2 and 3, respectively. All implementation parameters (number of configuration voltages, interpretation scheme, etc) used were kept constant across the control experiments. Important algorithms and search parameters are presented in Table 4.2.



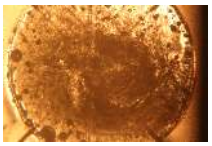


TABLE 4.2: Algorithms and Search Parameters.

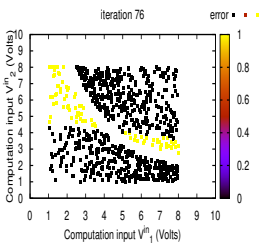
	Parameter	Value
DE	cross-over operator (CR)	0.7026
	differential weight (F)	0.814
PSO	constriction coefficient (ω)	0.0729, 0.729
	coefficients (c_1, c_2)	0.08, 0.9
	velocity $[\zeta_{min}, \zeta_{max}]$	[0, 1], [1, 4]
	information exchange	global
Search	iteration size (Λ)	150-300
	population size PSO (N)	10
	population size DE (N)	8
	$[V_{min}, V_{max}]$ (Volts)	[0, 4]
	$[R_{min}, R_{max}]$	[0.05, 15]
	$[p_{min}, p_{max}]$	[1, 182]
	scheme	$h^{(1)}$

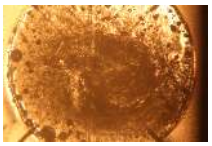
Both DE and PSO were used in the control experiments. The parameters of DE presented in the first row of Table 4.2 are those discussed in Chapter 3. On the other hand, the parameters of PSO presented in the second row are specific to this set of investigations, and therefore included here for the first time. The coefficients (ω, c_1, c_2) used in the global-PSO took two distinct values, depending on the configuration variable to update. For the configuration voltages and threshold, which can vary between [0, 4] Volts and [0.05, 15] respectively, the coefficient values used were $\omega = 0.0729$, $c_1 = c_2 = 0.08$ and $\zeta \in [0.0000001, 1]$. On the other hand, p can vary between [1, 182] and the coefficient values were therefore increased ($\omega = 0.729$, $c_1 = c_2 = 0.9$ and $\zeta \in [1, 4]$), allowing particles to take larger steps. This choice of values was based on results obtained during preliminary empirical investigations with SWCNT/LC. The data collected during these investigations is not included here.

The number of control experiments varied across the different combinations of material/EA/BCP, since for each control material, negligible variations in results were observed depending on the EA/BCP used. The discussion focuses on the SC problem, which was the most thoroughly investigated, with a minimum of three experiments per combination material/EA.

4.5.3 Results

For each material, representative examples of objective (error) function behaviour during training by DE and PSO are illustrated in Figure 4.3 (a) and (b), respectively. It can be

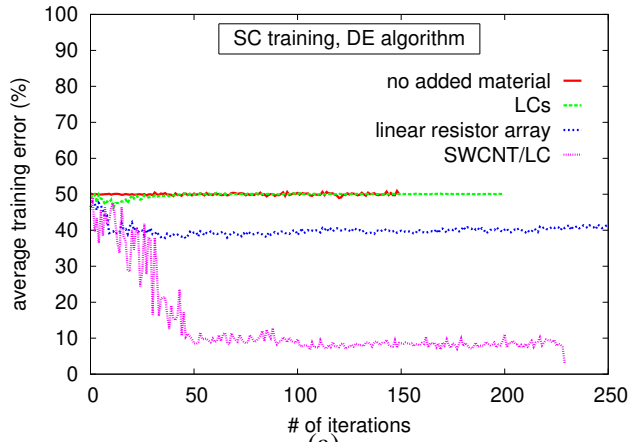




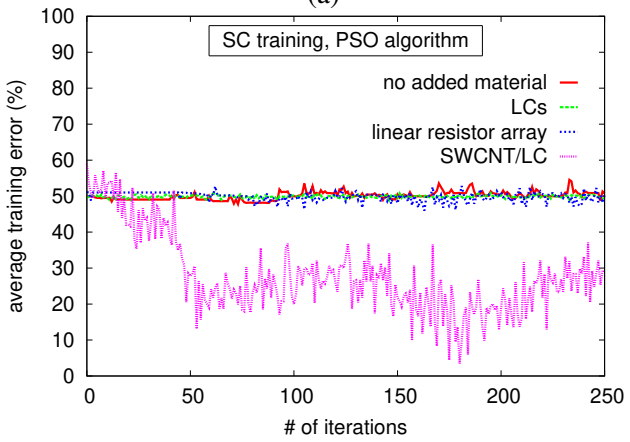
observed that neither algorithm was capable of finding a solution resulting in a perfect classification of all instances from the SC training dataset, i.e. an optimum of 0% error.

In the case of the empty micro-electrode array, i.e. no added material, the objective function remained around 50% for both EAs, which the worst possible solution for both BCPs, as it indicates a classifier unable to differentiate between classes in any other ways than by performing a random coin toss. The same level of error was observed during verification. Training did not modify the empty array's ability to classify instances from the SC dataset. Instead, they continued to be randomly classified throughout the experiment. This is coherent with the fact that, in the absence of added material between electrode terminals, the resistance is infinite. Measured currents should remain at 0A, irrespective of the inputs.

In practice, however, currents in the order of the pico-Amps were recorded by the *mbed* during training. They were found to be a measure of noise across the EM rather than the result of changes in conductivity across the empty array. The fact that the error

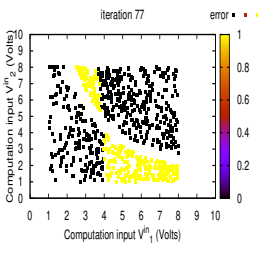


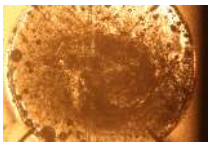
(a)



(b)

FIGURE 4.3: Convergence of the objective function, averaged per iteration, for three different control materials trained using the (a) DE and (b) PSO algorithms to solve the SC problem.





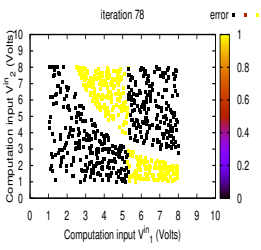
remained 50% suggests that the implementation does not create artificial solutions, i.e. a material allowing the algorithm to modify the current outputs needs to be present for solutions to be found.

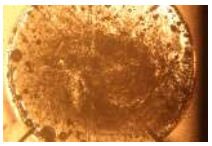
Drop-casting LCs across the array lowered the resistance between the terminals, but as discussed in [16], the currents remained negligible. The algorithms were unable to modify the classification error as can be observed in Fig. 4.3 (a) and (b) which show that the objective function remained around 50% throughout training. These results supports the suggestion that the system cannot find optimal solutions to the BCPs without an appropriate material (cf. Chapter 2). Moreover, it justifies the addition of SWCNTs to the LCs and the latter's contribution to the search space as a matrix allowing the SWCNTs to form conductive paths between electrodes [17].

Unlike the other control materials, the array of linear resistors enabled the finding of solutions minimising the classification error for the SC problem. It can be observed in Fig. 4.3 (a) and (b) that the objective function converges from 50% to around 35% error. When tested against unseen SC verification instances, the best possible solution found during training produced verification errors that remained around 35% ($\pm 3\%$) for all ten verification tests. These observations are consistent with [19], where results reported showed that an array of the same model could be exploited by the RCiM framework to solve computation problems. Here, however, despite the simplicity of the dataset, the error was far from the problem's minimum. The algorithms were not capable of finding the optimum, or even a good solution to the BCP when training the array of linear resistors.

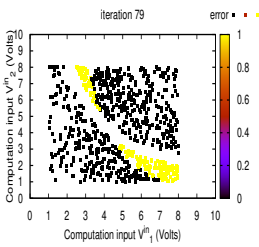
Finally, a convergence towards 0% error was observed in Fig. 4.3(a) for the SWCNT / LC sample. Replacing the control materials with SWCNT/LCs allowed DE to find an optimal solution to the training problem, which generalised well to the unseen data (1% verification error). Similarly, solutions classifying SC training and verification instances, with 0% and 1% error respectively, were found using PSO. This is not illustrated in Fig. 4.3(b), as large variations in the objective function during PSO's search, resulted in a disparity between average error per iteration (plotted in the figure) and the minimum error achieved per iteration. Differences based on the choice of algorithm are discussed in Section 4.7.

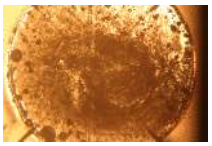
In summary, the ability of DE and PSO to solve the simple BCP was increased by





the use of SWCNT/LC samples. Using the same experimental implementation, optimal solutions could not be found with the resistor array, and the framework did not allow the algorithms to construct solutions from noise or negligible current levels. However, noise constitutes a bias which is part of the overall search space, and therefore part of any solutions obtained with SWCNT-based composites.





4.6 Comparing SWCNT concentrations

4.6.1 Motivations

Investigations reported in [12, 15] identified a critical 1 wt % SWCNT/PBMA concentration, for which search performances were better than for any other concentration investigated. It was observed that this concentration coincided with the percolation threshold of SWCNT/PBMA composites reported in [4].

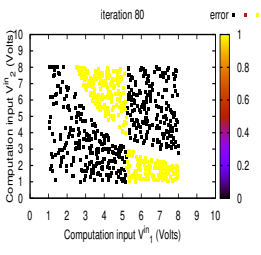
The following experiments first investigate whether the SWCNT concentration is a factor affecting training performance in SWCNT/LC composites. The aim is to identify, if it exists, an optimal concentration, or range of concentrations, for which the overall training performance results are best. The relation between this potential computationally optimum SWCNT concentration and the material's percolation threshold is also discussed.

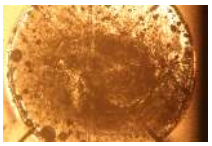
4.6.2 Experimental Implementation

Since similar relations between SWCNT concentration and computability were observed in preliminary experiments where DE and PSO were compared, only DE was used in the more extensive investigations presented here. Four different concentrations: 0.02, 0.05, 0.2 and 1 wt % SWCNT/LC were investigated, along with two BCPs. The 0.0025 wt % SWCNT/LC composite used in [17] was not included, as the classification errors obtained with this concentration were consistently higher than those obtained in the investigations reported here. The fully separable SC dataset and the partially merged MC dataset were chosen as they present different characteristics and complexity. For each combination of problem and concentration, five experiments were undertaken, using the implementation and the DE parameters presented in Table 4.2 of Section 4.5.2.

4.6.3 Results

Table 4.3 presents the best training error $\Phi_e^{t,*}$ and iteration λ_e^* at which it is reached, along with the difference between minimum training and verification $\overline{d_\Phi}$, the verification error $\overline{\Phi_e^v}$ and the standard deviation, $\sigma_{\Phi_e^v}$, which measures the variance in verification error across these experiments, i.e. the reproducibility of the process. Each metric is averaged over five experiments, and the best results are reported in bold. In addition, the spread of





verification error across experiments for the four different SWCNT/LC concentrations and the two BCPs is illustrated in Figure 4.4. The median, interquartile range and outliers for $\Phi^{v}_{e,j}$ are reported in a box plot format.

Overall, results show that EiM training was most efficient when implemented in 0.05 wt % SWCNT/LC samples. Whilst this concentration was second best in terms of training error and speed of convergence for the SC problem, it performed better than the other concentrations in all other efficiency metrics reported, for both BCPs. In terms of verification errors, optimum values were achieved in at least one of the experiments, as illustrated by the lower bound of the interquartile range in Figure 4.4. On average, the verification error was close to 0% in the case of SC and 3.3% in the case of MC, which are these problems' respective optimums. Finally, compared to the other concentrations, solutions found with the 0.05 wt % SWCNT/LC samples generalised better overall and produced consistently low verification error. These two observations are illustrated by the fact that the lowest values of $\overline{d_{\Phi}}$ and $\sigma_{\Phi^{v}_{e}}$, respectively, were achieved by this composite. The size of the interquartile range and number of outliers reported in Figure 4.4 also give an indication of the reproducibility of the training. It can be observed that the

TABLE 4.3: Training and verification errors for different SWCNT/LC concentrations.

	SC				MC			
wt %	0.02	**0.05**	0.2	1	0.02	**0.05**	0.2	1
$\Phi^{t,*}_{e}(\%)$	**0.000**	0.780	4.460	17.120	5.760	**4.42**	6.820	15.025
λ^{*}_{e}	**21.6**	136.8	274.4	252	285.5	**182.8**	183.4	261.6
$\overline{d_{\Phi}}(\%)$	3.455	**0.235**	1.455	3.030	10.175	**0.995**	14.795	1.981
$\overline{\Phi^{v}_{e}}(\%)$	4.773	**1.241**	7.660	21.803	17.448	**5.686**	22.415	15.841
$\sigma_{\Phi^{v}_{e}}$	1.257	**1.022**	5.367	12.347	7.317	**1.819**	16.803	8.667

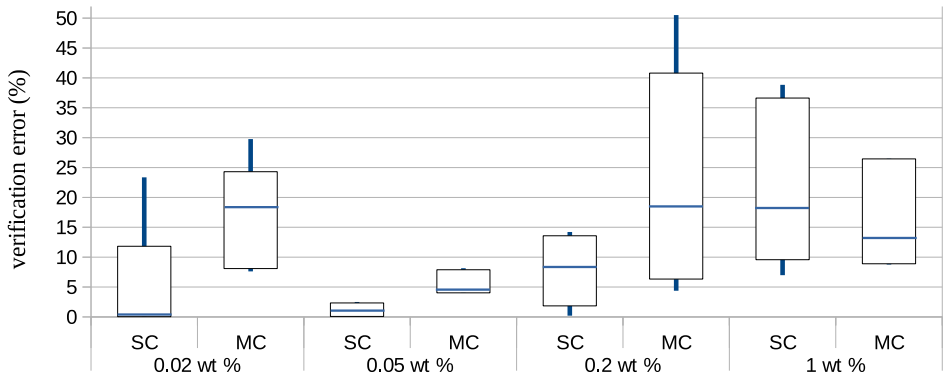
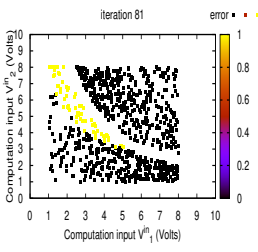
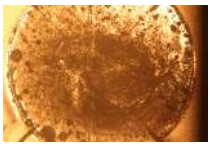


FIGURE 4.4: SC and MC verification error for the four different SWCNT concentrations, in terms of minimum and maximum values, inter-quartile range and median.



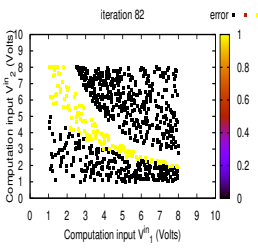


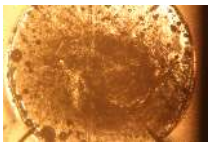
0.05 wt % SWCNT/LC solutions have the smallest interquartile range, and over the five experiments, no outliers were obtained.

Across both BCPs, the higher concentration composites, 0.2% and 1%, tended to be better at generalising than the 0.02% composite, but the training and verification errors were high and the spread of verification error across the mean important. On the other hand, the lowest concentration tended to find good solutions during training, resulting in low training error, and the verification results were relatively reproducible. However, the solutions did not generalise well. In other words, a low training error did not always coincide with low generalisation. For example, in the case of the MC problem, the 0.02 wt % SWCNT/LC samples had $\Phi_e^{t,*} = 5.760$ whilst the 1% samples averaged $\Phi_e^{t,*} = 15.025\%$, but the difference between best training and best verification errors were $\overline{d_\Phi} = 10.175\%$ and $\overline{d_\Phi} = 1.981$, respectively.

These observations suggest that SWCNT concentration has an impact on both classification accuracy and result reproducibility. The photographs taken throughout training suggest an effect of the concentration on the change in morphology induced in the material throughout training. A large amount of variations were observed in the 0.02 wt % SWCNT/LC composite, from iteration to iteration, whilst next to none were observed in the composites with 0.2 and 1 wt % SWCNT concentration. Figure 4.5 shows representative examples of photographs taken in the different concentrations, one during the first iteration, and one during the last iteration of training for the MC problem. The SSIM reported in this figure is between these two iterations only, i.e. it indicates the overall change in the observable morphology of the samples, rather than the RoC they experienced throughout training.

The differences in the level of change in morphology can partly explain the differences in classifier performance across concentrations. SWCNTs tend to aggregate under the influence of Van der Waals forces. At lower SWCNT concentrations, the percolation paths, formed by the SWCNT aggregating into bundles across the composite, are more likely to vary throughout training, due to the higher movement observed across the material. This allows solutions to be found irrespectively of the composite's initial state (unlike solid SWCNT/polymer composites), as illustrated by the lower average verification error across experiments reported in Table 4.3. However, it also means that the evolved percolation paths forming part of the optimum solution found during training





have a high chance of being modified by subsequent voltage application, material drift or external factors. In other words, around 0.02 wt % SWCNTs, solutions produced with SWCNT/LC samples were not stable.

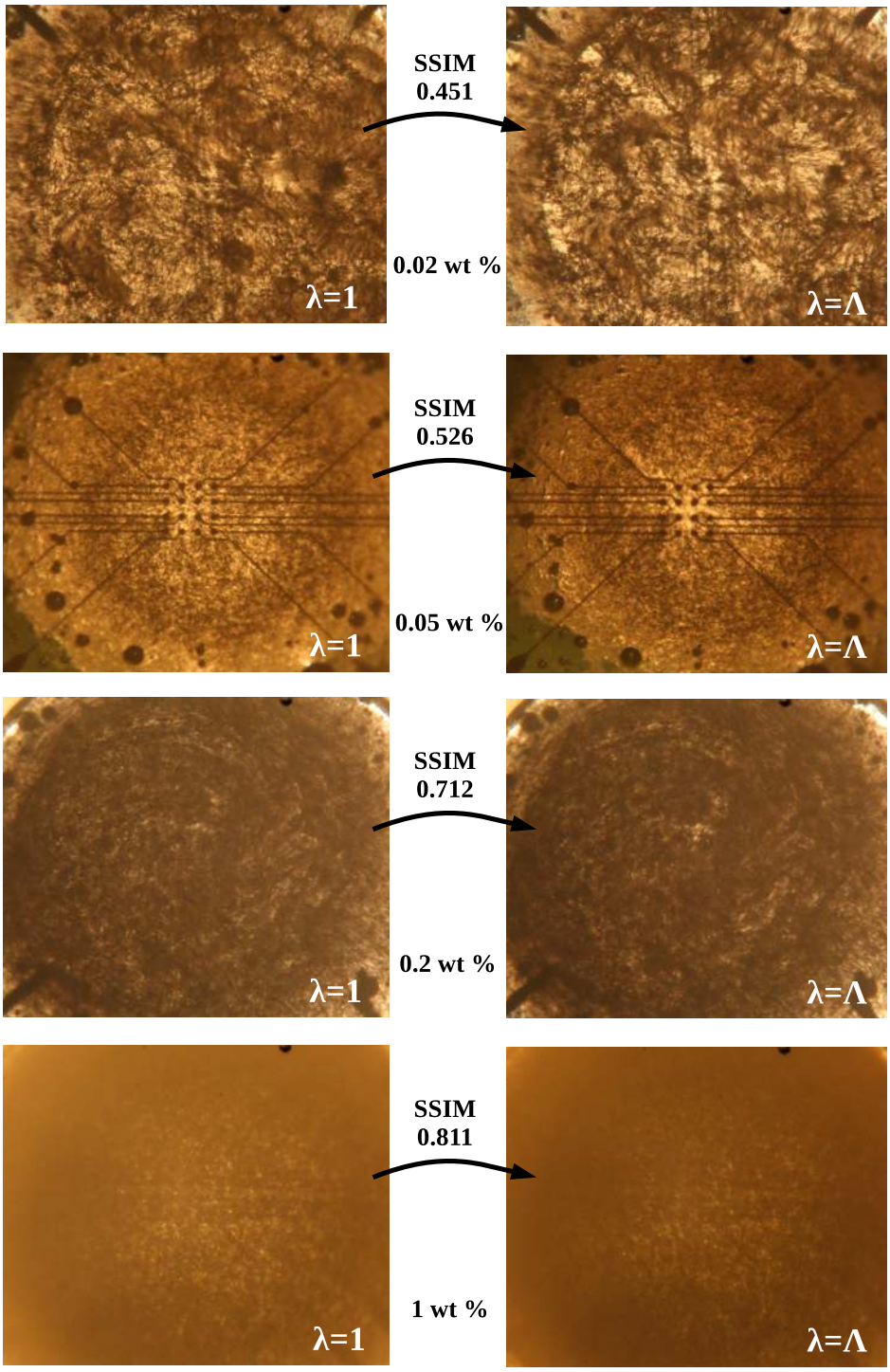
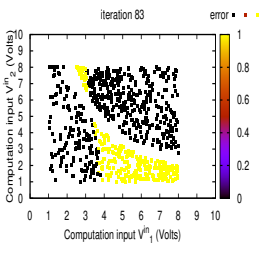
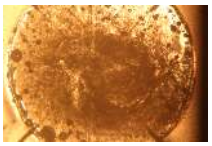


FIGURE 4.5: Photographs of the SWCNT/LC composites' surface taken at the start ($\lambda = 1$) and the end ($\lambda = \Lambda$) of training performed by the differential evolution algorithm to solve the MC classification problem. The photographs are arranged vertically by increasing SWCNT concentration. A high SSIM value means small amount of perceptible change in material morphology first and last iteration of training. The SSIM is seen to increase with SWCNT concentration.





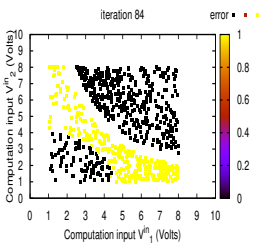
Following this line of reasoning, results obtained at higher concentration can be explained by the fact that it is harder to modify SWCNT bundles formed in the samples prior to training. Experimental results were therefore more dependent on the material's initial state and whether or not it favoured the finding of good solutions to the classification problems. The optimum concentration, where solutions were able to generalise better and classify instances more accurately combined a viscosity that allowed the SWCNTs to move under an applied electric field, but not so much as to destroy an evolved problem-solving material state ($\mathbf{M}^*$).

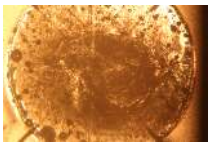
In summary, optimum training speed and classification accuracy were obtained for both BCPs when SWCNT/LC composites had concentrations of 0.05 wt % SWCNTs. In addition, evaluation of the RoC throughout training supported the hypothesis that the EA's search had an effect on the material morphology. The photographs taken with the microscope were clearer for 0.05 wt % SWCNT/LC samples than with the two higher concentrations, enabling optical changes in the material to contribute to the analysis of results, along with classification errors, and electrical input/output variations. It must be noted, however, that for all concentrations, the photographs only captured movements from the samples' top surface, and at the micrometer level. The hypothesis formulated here would be better supported using higher resolution microscopes or microscopes collecting data across three dimensions.

4.7 Comparing Evolutionary Algorithms' Performance

4.7.1 Motivations

Within the field of EiM, genetic algorithms (GA) and evolutionary strategies (ES) have been the favoured choice of EA, but no investigations into their relative performance has been reported. On the other hand, the Nelder-Mead (NM) and DE algorithms have been compared in [20], along with PSO in [3]. Non-negligible differences in search behaviour and result accuracy were reported in both, showing a non-negligible dependence of results on the choice of algorithm. These observations motivated the investigations presented in this section, especially since both the material and the problems used differ from the solid SWCNT/polymer composites used in previous discussions, allowing potentially different conclusions to be drawn.





4.7.2 Experimental Implementation

The performance of DE and PSO algorithms were compared using samples of 0.05 wt % SWCNT/LC composite. The algorithm and search parameters are the same as those presented in the table 4.2 of section 4.5.

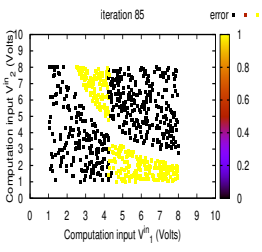
The comparisons focus on the search performance of the EAs resulting from their training of the SWCNT/LC composite to solving the SC, MC, V1C and NLC problems. The search terminates either if an error within 0.5 % of the problem's optimum has been obtained repeatedly over two iterations, resulting in a variance in training error $< 0.5\%$, or if the search has reached the maximum number of iterations $\Lambda = 350$.

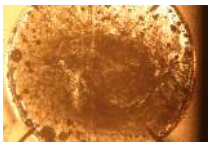
For each problem and algorithm, a minimum of $W = 5$ experiments were undertaken. As part of each experiment, ten verification tests were used to verify the accuracy of the SWCNT/LC classifier subjected to DE or PSO training. Verification tests started 300s after training ended. For each test, the same set of verification instances was applied to the material along with the optimum set of decision variables obtained during training. Observable changes in the samples' SWCNT structures were recorded during training using a camera fixed to a microscope.

4.7.3 Results

The first column of Table 4.4 presents the minimum error $\Phi_e^{t,(\lambda^*,\ell^*)}$ achieved during training, which corresponds to the error produced by the solutions $\mathbf{x}^*$, by individual ℓ^* at iteration λ^*. The notation $\Phi_e^{t,*}$ is for the averaged error over the W experiments undertaken. The other four columns of Table 4.4 refer to results of the verification tests: $\Phi_e^{v,*}$ is the minimum verification error, $\Phi_e^{v,w}$ the worst, $\overline{\Phi}_e^{v}$ the average and $\sigma_{\Phi_e^v}$ the standard deviation from this average across experiments. The latter gives an idea of the stability of the solution, rather than the reproducibility of the results obtained with one or the other algorithm.

It must be noted that due to the state of the material, if the optimum solution $\mathbf{x}^*$ producing the minimum error was achieved on the last iteration ($\lambda^* = \Lambda$), then the first verification test, t_1, is applied to $\mathbf{M}^*$, i.e. the optimum solution $\mathbf{x}^* = [\mathbf{V}^*\mathbf{R}^*\mathbf{M}^*]^T$ is tested against unseen data. All subsequent tests are applied on a material which is no longer in the exact state $\mathbf{M}^*$. However, if $\mathbf{x}^*$ was obtained earlier in the training sequence, none of the tests are applied on a material in the exact state $\mathbf{M}^*$. In that case,





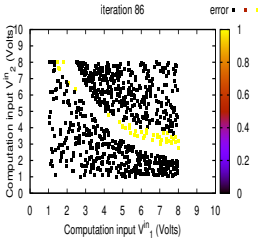
the verification tests are used to assess 1) the stability of $\mathbf{M}^*$ or 2) the robustness of the part of the solution $\mathbf{x}'^* = [\mathbf{V}^*\mathbf{R}^*]^T$, i.e. their ability to set the material in a problem solving state, irrespective of physical changes in $\mathbf{M}^*$.

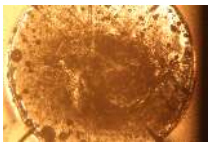
The difference between training and verification errors, excluding the outliers, remains $|\Phi_e^{t,*} - \Phi_e^{v,*}| < 2.125\%$ across the different problems and algorithms. This indicates that the material's behaviour is consistent and generalises well as a classifier. A poor generalisation would see the difference between training and verification increase towards 50%. Solutions obtained during training using DE can be better than those of PSO, especially for the SC and V1C datasets. However, PSO outperforms DE with respect to consistency across experiments and generalisation of the solution. This can be observed for the MC dataset where DE obtains both smallest and largest error for verification ($\Phi_e^{v,*} = 3.975\%$ and 18.25% respectively) whilst PSO's variance over verification tests is lower.

Figure 4.6 illustrates the convergence of the error during training in terms of average and minimum per iteration, for DE and PSO and based on the four BCPs, SC, V1C, MC and NLC. The convergence patterns observed are representative of the experiments in Table 4.4, irrespective of the experiment undertaken. In addition, the objective function produced during one of the control experiments is illustrated in the figure, for the sake of comparison. The control material consists in the LCs drop-cast on the electrode array and the values reported in each graph corresponds to the values obtained when this control material is trained using the same combination of problem and algorithm as

TABLE 4.4: Training and verification errors for SC, MC, V1C and NLC problems.

SC Experiments	$\Phi_e^{t,*}(\%)$	$\Phi_e^{v,*}(\%)$	$\Phi_e^{v,w}(\%)$	$\overline{\Phi_e^v}\,(\%)$	$\sigma_{\Phi_{e,v}}$
PSO	1.400	1.925	2.667	2.385	0.071
DE	4.233	6.350	7.558	6.995	0.154
MC Experiments	$\Phi_e^{t,*}(\%)$	$\Phi_e^{v,*}(\%)$	$\Phi_e^{v,w}(\%)$	$\overline{\Phi_e^v}\,(\%)$	$\sigma_{\Phi_{e,v}}$
PSO	5.567	6.917	8.300	7.712	0.261
DE	5.167	3.975	18.250	10.454	0.117
V1C Experiments	$\Phi_e^{t,*}(\%)$	$\Phi_e^{v,*}(\%)$	$\Phi_e^{v,w}(\%)$	$\overline{\Phi_e^v}\,(\%)$	$\sigma_{\Phi_{e,v}}$
PSO	2.7	3.975	5.175	4.653	0.1318
DE	1.3	2.325	2.725	2.498	0.016
NLC Experiments	$\Phi_e^{t,*}(\%)$	$\Phi_e^{v,*}(\%)$	$\Phi_e^{v,w}(\%)$	$\overline{\Phi_e^v}\,(\%)$	$\sigma_{\Phi_{e,v}}$
PSO	0.06	4.275	7.325	6.060	0.845
DE	1.40	23.175	23.825	23.513	0.0384





the SWCNT/LC sample it is compared against in the plot, i.e. in Figure 4.6 (a), both SWCNT/LC and LC-only samples are trained using DE to solve the SC problem.

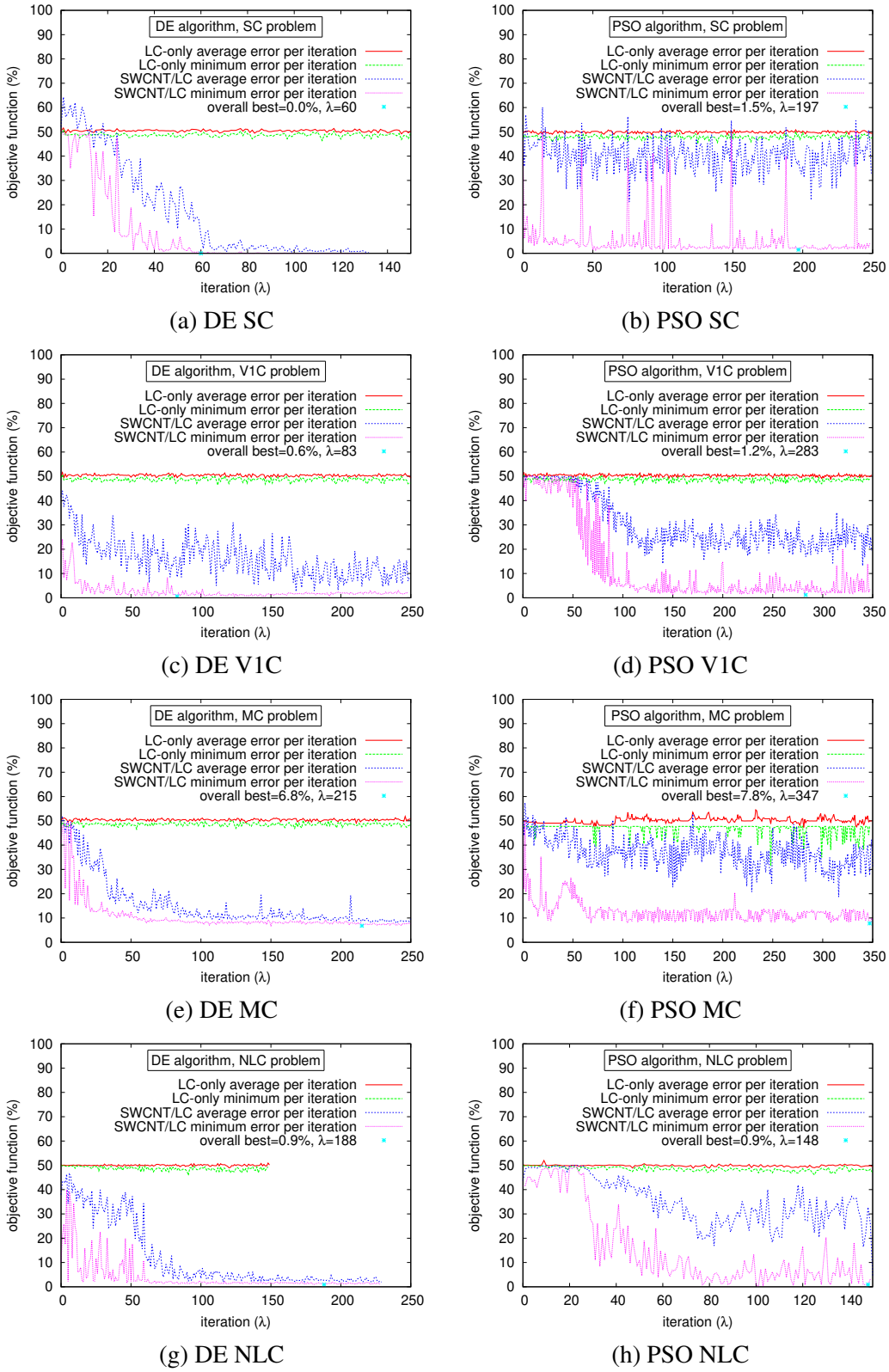
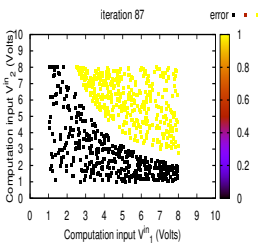
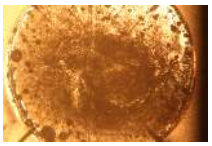


FIGURE 4.6: Convergence of the objective function produced by DE and PSO during representative trainings of SWCNT/LC samples for all the synthetic binary classification problems.

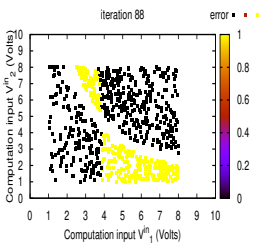


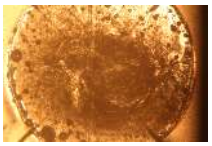


It can be observed from the plots presenting the average and minimum values of the objective function per iteration throughout training in Figure 4.6 (a), (c), (e) and (g), that the SWCNT/LC sample adapts within few iterations when DE is used. Once an optimum solution, resulting in a % of error close to the problem's minimum, has been reached, the algorithm spends the subsequent iterations exploiting solutions around this optimum. This results in a convergence of the average and minimum errors towards the same value, around which they revolve until the last iteration, or a suitable termination criterion has been reached. On the other hand, the PSO algorithm reaches errors close to the problems' optimum towards the end of the training process, exploring the overall search space. This is suggested by the large difference between average and minimum error averaged per iteration observed in Figure 4.6 (b), (d), (f) and (h), i.e. all the right hand graphs, which present the convergence of the objective function obtained with the PSO algorithm for the four BCPs.

Following the training phase, verification tests were performed on the evolved devices. Two examples of the resulting distribution of misclassified verification instances, obtained from devices where DE and PSO converged to solutions producing the same minimum training error $\Phi_e^* = 5.7\%$, are presented in Figures 4.7 (a) and (b). Results are for one of the ten verification tests only. In both plots, the overlapping area between the two classes contains the majority of the misclassified instances. However, outside the overlap, less instances are misclassified by the PSO-trained device than by the one trained using DE. The percentage of verification error presented in Figures 4.7 (a) and (b) were 18.475% and 7.825 %, respectively. Despite both algorithms achieving the same training error, DE misclassified 426 instances more than PSO.

Figures 4.7 (c) and (d) show the distinctive difference between the two algorithm's configuration voltages' trajectories, averaged over S, per iteration. The search performed by DE is more noisy even when the algorithm aims to exploit a minimum. On the other hand, PSO's exploration of the search space is based on smoother inputs. This is the expected behaviour of particle trajectories for the PSO algorithm [21]. This difference has been proposed in [22, 23] as a possible explanation for the fact that over a number of experiments, DE is less consistent in its performance, whilst exploration of the search space by the PSO algorithm tends to produce devices with a classifying state that can be recovered post-training, and which generalises well to new data . It is possible that the





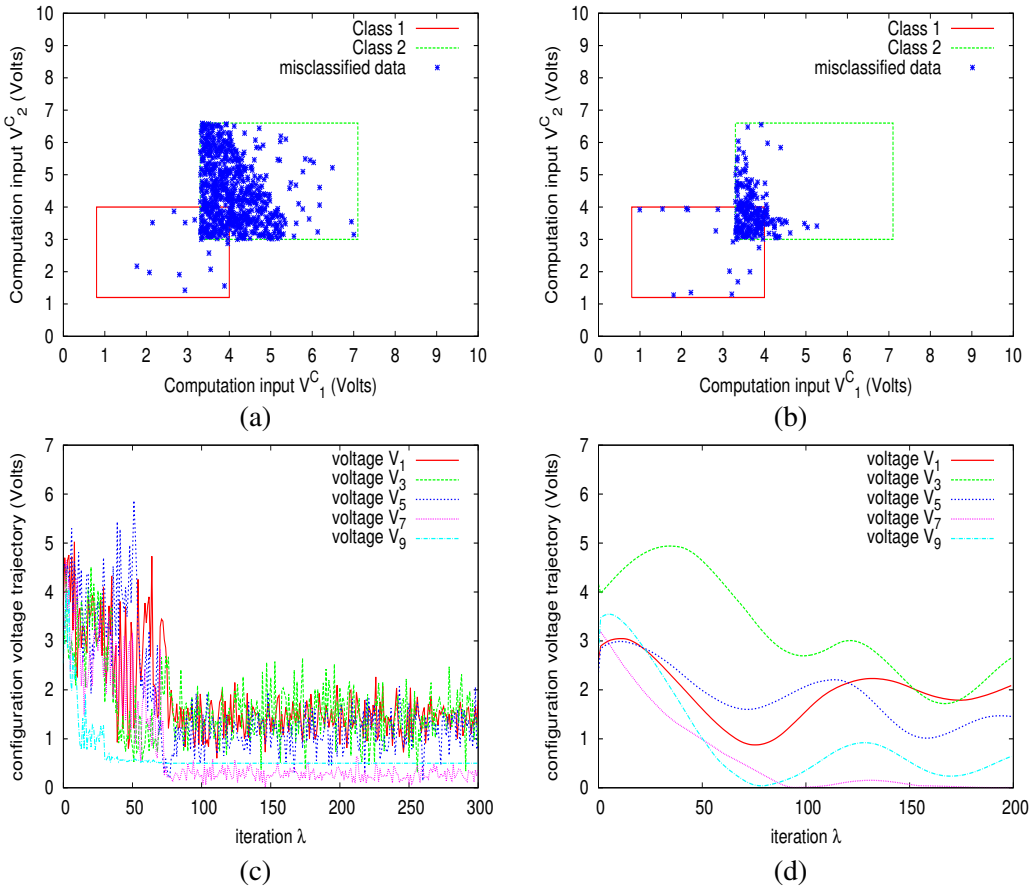
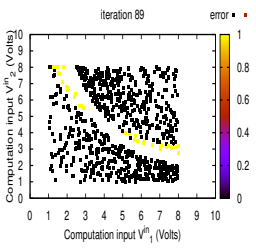


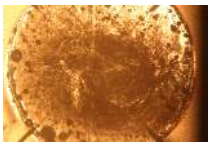
FIGURE 4.7: Visualisation of the classification error achieved during verification for (a) DE and (b) PSO, followed by the configuration voltage trajectories produced by (c) DE and (d) PSO.

PSO algorithm's smoother trajectories of V_j allow stable SWCNT structures reinforcing responses minimising the classification error to be built inside the material. The noisy V_j applied by DE would make the formation of such structures more difficult.

It must be noted, however, that the differences reported are based on bulk composite measurements. Another interpretation based on the same measurements could suggest that the generalisation property depends on the overall level of the electric field applied to the samples. DE-controlled voltages tend to converge towards low values early in the search, whilst PSO's remain close to half of the maximum voltage allowed until later in the search. Knowing the exact changes in the SWCNT-distribution across the LC matrix throughout training would provide more information regarding the impact of the two different algorithms on the material, and perhaps support one or the other hypothesis. It is an area that remains to be explored beyond the scope of this thesis.

Figure 4.8 (a) depicts the convergence trajectory of p all experiments where DE and PSO are used to train 0.05 wt % SWCNT/LC samples into solving the MC problem. Convergence is not towards the same value of p, but resulting input electrode location





are similar for both algorithms. Figure 4.8 (b) presents the corresponding mapping of p with regard to input location on the micro-electrode array for the optimal solutions of the three problems. Experiments resulting to errors between 4-10% for $\Phi_e^{t,*}$ and $\Phi_e^{v,*}$ tend to have a p corresponding to the most favoured locations shown in Figure 4.8 (b). It must be noted that two close values of p might not correspond to two adjacent electrode positions. The convergence is therefore towards a target set of electrodes, but each small variation from the value of p resulting in this target or specific electrode assignment might result in a large change in electrode location for the problem's inputs and the algorithm's configuration voltages.

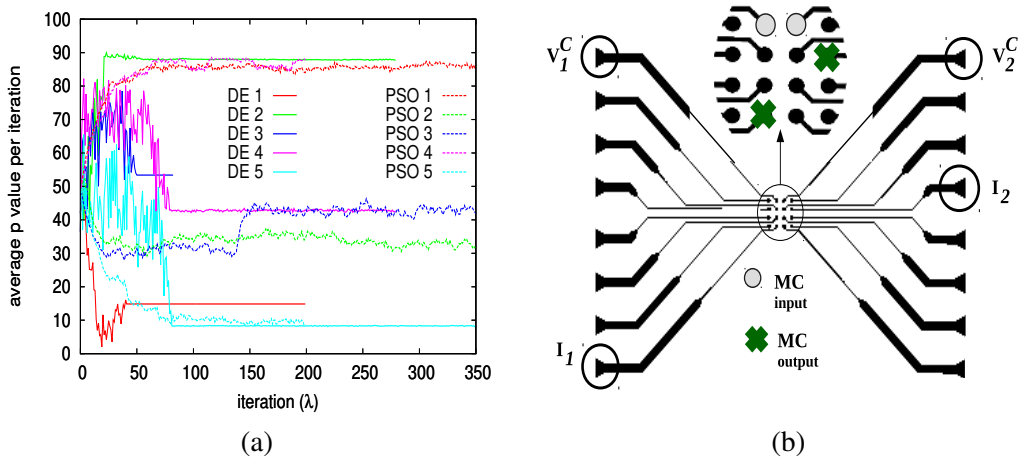
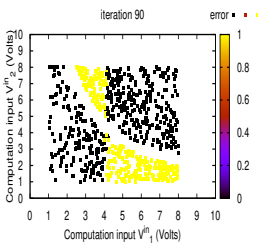


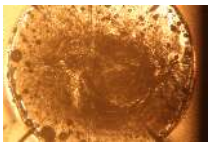
FIGURE 4.8: (a) Visualisation of the evolution of p controlled by DE and by PSO to solve the MC problem. In (b), the value of p^* obtained in these experiments is translated into the most common electrode positions.

4.8 Comparing Problem Formulation Parameters

4.8.1 Motivations

Similarly to the choice of algorithm, the choice of problem formulation has been observed to have an impact on the material's ability to solve the BCPs [19]. Since the origin of this impact in not currently known, problem formulation parameters tend to be chosen arbitrarily, or ansatz. This is the case for the simple function, $h^{(1)}$ from eq. (4.7), used in the interpretation scheme and developed to solve simplest classification problems. The aim here is therefore to investigate whether better results can be obtained from variations in search parameters and problem formulation for the SWCNT/LC composite and the algorithms previously described. A discussion based on robust statistical





analysis of the data collected would provide a basis for the choice of search parameters, enabling subsequent investigations into the solving of more complex datasets.

4.8.2 Experimental Implementation

Varying search parameters and variables

Investigating the impact of the search parameters and variables on the computational results focused mainly on an analysis of the variable p, but experiments also involved:

- varying the maximum and minimum values of the threshold used in the interpretation scheme, i.e. R_{max} between 5 to 30, and R_{min} from 0 to 0.05
- varying the maximum configuration voltages level, V_{max}, between 2V and 8V.

Varying the function used in the interpretation scheme

A detailed description of the problem formulation used to transform an un-configured material into a classifier is given in Chapter 3. Errors obtained when training SWCNT/LC samples using four different problem formulations are compared in this section.

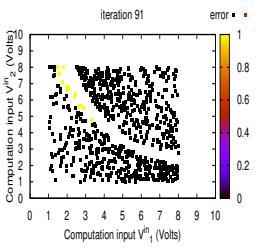
The first form of the function h used in the problem formulation was $h^{(1)}$, presented in section 4.2.2 and yielding interpretation scheme $S_{\mathcal{C}}^{(1)}$. A variation on this function involved multiplying the output currents collected across the material by the computation voltages defining the instances to be classified, resulting in $h^{(2)}$ of the form:

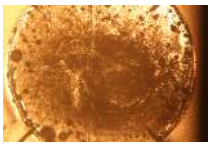
$$h^{(2)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}(\mathbf{M})\right) = h^{(2)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{Y}(\mathbf{M})\right) = I_1 V_1^{\mathcal{C}} + I_2 V_2^{\mathcal{C}} \tag{4.12}$$

yielding interpretation scheme $S_{\mathcal{C}}^{(2)}$. The $S_{\mathcal{C}}^{(2)}$ combines information collected from both the material and the observable data generating source. In principle, a different h depending explicitly on $\mathbf{V}$ can also be used, as is the case in [20]. Instead, the dependence in (4.12) is implicit through the physical quantities recorded across the material, $\mathbf{Y}(\mathbf{M}) = [I_1(M), I_2(M)]^T$, which are measures of the configuration voltages, in addition to the material state and computation voltages.

A second variant to the original function involved the same multiplication between inputs and outputs used in h^2, but in this case the currents I_1 and I_2 were raised to the third power such that

$$h^{(3)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\right) = h^{(3)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{Y}\right) = (I_1)^3 V_1^{\mathcal{C}} + (I_2)^3 V_2^{\mathcal{C}} \tag{4.13}$$





is the function used in the cubic scheme $S_{\mathcal{C}}^{(3)}$.

The last variant involved a logistic function, or sigmoid, which is a common type of activation functions used to regulate the output of a neuron within an artificial neural network (ANN). In the ANN case, the sigmoid can be used in the hidden layers of the network as well as within the output layer, for example to infer a class from the set of inputs sent to the network when the ANN is trained to behave as a data classifier [24]. Here, similar to the latter use, the class of a data-point is inferred from the material's outputs using a sigmoid of the form

$$h^{(4)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\right) = h^{(4)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{Y}\right) = \frac{1}{1+e^{-A}}, \tag{4.14}$$

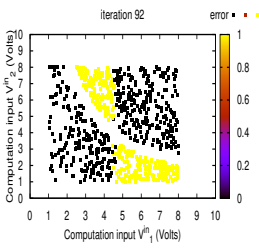
where A is a function of the output currents measured across the material and two decision variables, R_1 and R_2, such that

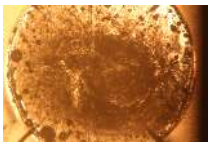
$$A = R_1 I_1(k) + R_2 I_2(k). \tag{4.15}$$

In this case, $h^{(4)}$ was compared to 0.5 rather than to an algorithm controlled variable, resulting in the $S_{\mathcal{C}}^{(4)}$ scheme:

$$S_{\mathcal{C}}^{(4)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}, \mathbf{R}\right) = \begin{cases} 1 & \text{if} \quad h^{(4)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\right) \leq 0.5 \\ 2 & \text{if} \quad h^{(4)}\left(\mathbf{V}^{\mathcal{C}}, \mathbf{V}, \mathbf{Y}\right) > 0.5 \end{cases} \tag{4.16}$$

There are alternatives to building the objective function by simple addition of the number of misclassified datapoints. One such alternative is to use false positives (FP), false negatives (FN), true positives (TP) and true negatives (TN) to build the error function. This scheme is commonly used when evolving classifiers to solve multi-class problems, or binary-class problems with class imbalance, as it gives a better indication of the classifier's true accuracy. For example, if 80% of a problem's instances belong to one class and 20% to another, a 80% training accuracy actually means that the classifier is randomly classifying data. Such a false sense of accuracy is prevented by the confusion matrix constructed from the FP, FN, TP and TN values. The use of confusion matrix in the scheme can also be useful when medical datasets are studied, as it provides further indication regarding the instance's class assignment: one might want to focus on maximising the number of true positives, for obvious reasons, but it is also useful to





minimise the amount of false positives, as being wrongly diagnosed with a disease can create unwanted level of stress. In the field of EiM, the scheme has been implemented when investigating the ability of training data classifiers [1, 15], frequency classifiers [25] or even parity problem solver [26] based on solid SWCNT/polymer composites. Following this work, the FP, FN, TP and TN are evaluated for each k instance of the training or verification dataset as

$$\begin{aligned} &TP(k+1) = TP(k) + 1, \quad TN(k+1) = TN(k) + 1 \quad \text{if} \quad C_M(\mathbf{V}^{in}(k), \mathbf{x}) = C(\mathbf{V}^{in}(k)) \\ &FP(k+1) = FP(k) + 1, \quad FN(k+1) = FN(k) + 1 \quad \text{if} \quad \text{otherwise.} \end{aligned} \tag{4.17}$$

The mean total error in this case is given by

$$\Phi_e^{(TP)}(\mathbf{x}) = \sum_{k=1}^{K_t} \frac{(TP(k)TN(k))}{(TP(k) + FP(k))(TN(k) + FN(k))}. \tag{4.18}$$

and the optimisation problem for this scheme, $S_{\mathcal{C}}^{(5)}$, becomes

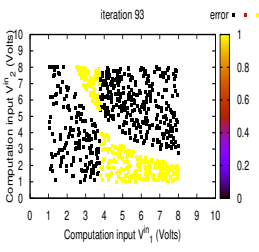
$$\begin{aligned} &\min_{\mathbf{x}} \; -\Phi_e^{(TP)}\left(\mathbf{x}, \mathbf{V}_t^{\mathcal{C}}, K_t\right) \\ &\text{subject to (3.13)} - \text{(3.15)} \end{aligned} \tag{4.19}$$

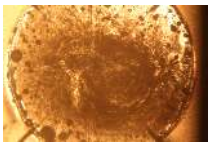
4.8.3 Results

Varying search parameters and variables

Different values of the search parameters were tested in preliminary investigations, meaning that the number of experiments undertaken with different values is too small to be reported in a table (generally below 3 experiment per parameter value). However, some cases showed important variations between the few experiments undertaken, and led to the following observations and choice of values:

- Better training and verification results, i.e. combination of smallest errors and fastest algorithm convergence, were achieved for $R_{i,\max} = 15$ rather than for any other of the tested values. $R_{i,\min}$, $i = 1, \ldots n_2$ was first set to 0. In this case, the voltages controlled by the EAs tended to go to 0 after a few iterations, whether or not error had reached an optimum value. This was no longer the case when all $R_{i,\min} = 0.05$.





- $V_{i,\max} = 4V$ tended to result in the best solutions. It was observed that below this level, no change in the morphology of the SWCNT/LC samples during training tended to occur, whilst above 4V, these changes tended to be very important throughout evolution of the device.

It was also observed experimentally that the input voltage location variable, p, is important to the solution. Over a majority of experiments, the error function converges rapidly to an optimum once p has settled to a specific value. The voltages around that iteration will also settle and start to explore the search space around given values. In addition, the preferred choice of input location on the electrode array, over 20 experiments, varies according to the problem at hand, which means it is not random.

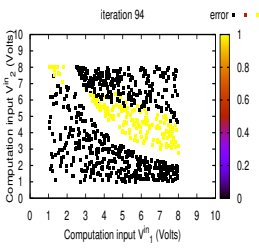
Varying the function used in the interpretation scheme

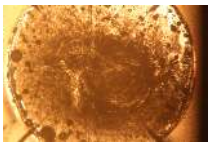
The first scheme, $S_{\mathcal{C}}^{(1)}$, depends only on the measured material responses. This proved sufficient for SC and MC, leading to the use of this variant in subsequent experiments. However, as the problem complexity increased, an increase in training and verification errors suggested the need for a more complex threshold function. The second scheme, $S_{\mathcal{C}}^{(2)}$ based on the function variant $h^{(2)}$ demonstrated significant improvements in the ability of DE and PSO-trained SWCNT/LC to solve the V1C and NLC problems.

The most complex of the BCPs, NNLC, could not be solved using either $S_{\mathcal{C}}^{(1)}$ or $S_{\mathcal{C}}^{(2)}$, directing investigations where $S_{\mathcal{C}}^{(3)}$, $S_{\mathcal{C}}^{(4)}$ were used instead. Training and verification results used to compare the different schemes are presented in Table 4.5. Results obtained with the objective function variant defined in eqs. (4.17)-(4.19), yielding $S_{\mathcal{C}}^{(5)}$, are also reported for comparison with the other alternatives.

TABLE 4.5: Training and verification errors for NNLC problems using different schemes.

Scheme	$\Phi_e^{t,*}(\%)$	$\overline{d_\Phi}$	$\overline{\Phi_e^v}(\%)$	$\sigma_{\Phi_e^v}$
$S_{\mathcal{C}}^{(1)}$	7.90	13.05	21.26	7.108
$S_{\mathcal{C}}^{(2)}$	5.9	1.63	8.14	0.091
$S_{\mathcal{C}}^{(3)}$	6.4	7.05	19.85	2.623
$S_{\mathcal{C}}^{(4)}$	15.37	11.11	26.99	11.767
$S_{\mathcal{C}}^{(5)}$	14.55	9.8125	19.92	7.703





Best results for this problem were obtained with the $S_{\mathcal{C}}^{(2)}$ schemes. However, it can be seen in Figure 4.9 (a) that the optimum solution obtained at the end of the evolution for the NNLC's verification datasets is separating the classes with a straight line. In the case of the $S_{\mathcal{C}}^{(3)}$ scheme, the line becomes non-linear, but does not fit the classes' boundary (fig. 4.9 (b)). It would be expected that the more sophisticated schemes, $S_{\mathcal{C}}^{(4)}$ and $S_{\mathcal{C}}^{(5)}$, would help produce a better fitting separation curve between the classes. Instead, they do produce a solution where the data is separated by a straight line far from the classes' boundary in the case of $S_{\mathcal{C}}^{(4)}$ (fig. 4.9 (c)) or a hyperbola in the case of $S_{\mathcal{C}}^{(5)}$ (fig. 4.9 (d)). The results are worse than for the simpler $S_{\mathcal{C}}^{(2)}$ and $S_{\mathcal{C}}^{(3)}$, with 26.99% and 19.982% verification error for $S_{\mathcal{C}}^{(4)}$ and $S_{\mathcal{C}}^{(5)}$, respectively.

Results for the NNLC dataset were inconclusive, in the sense that, irrespective of the scheme used to produce the objective function, the DE algorithm was unable to find the

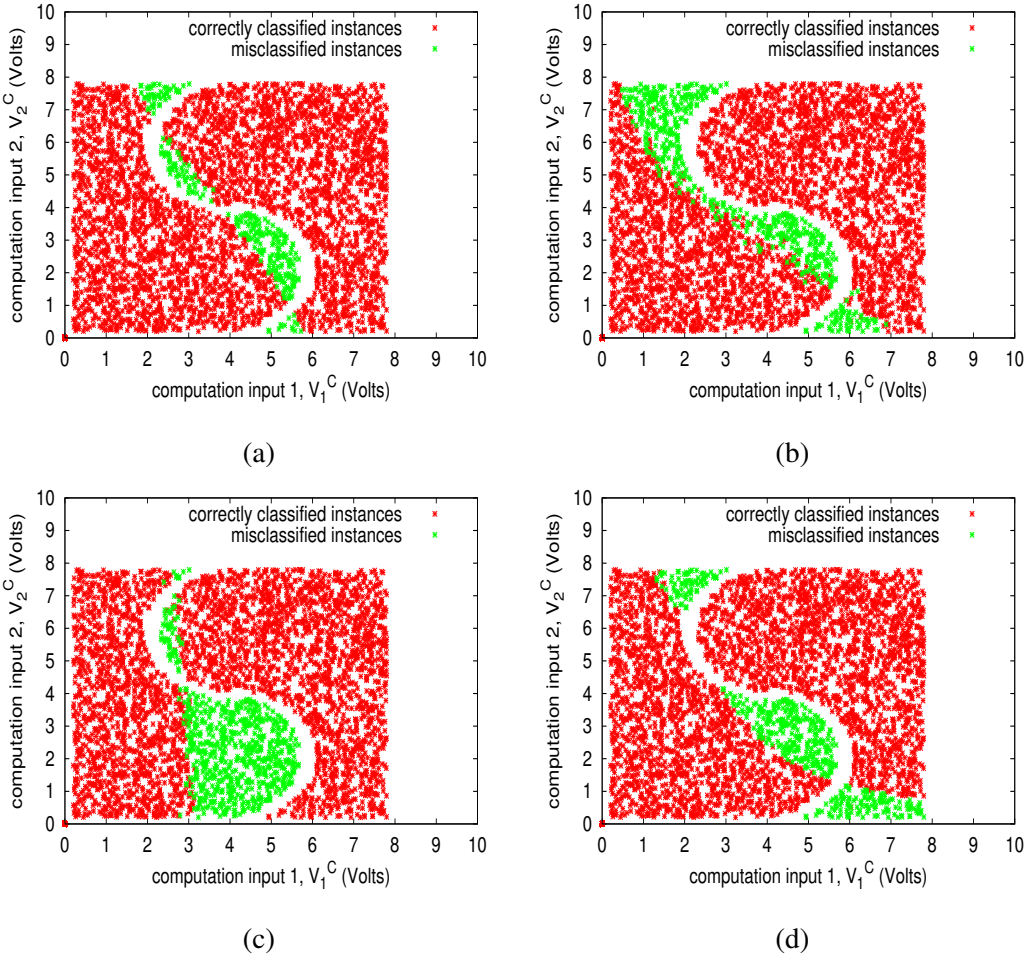
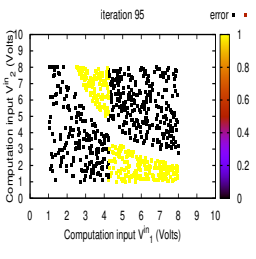
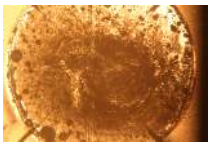


FIGURE 4.9: Verification results for NNLC dataset using (a) the non-linear scheme $S_{\mathcal{C}}^{(2)}$ from eq. 4.12, (b) the combined cubic scheme $S_{\mathcal{C}}^{(3)}$ from eq. 4.13, (c) the sigmoid-based scheme, $S_{\mathcal{C}}^{(4)}$, from eqs. (4.14)-(4.16) and (d) the TP-scheme, $S_{\mathcal{C}}^{(5)}$ from eqs. (4.17)-(4.19)





problem's optimum solution. Being fully separable, 0 % error should be obtained, but the minimum verification error over all experiments, and verification tests, is 7.38%.

It must be noted that due to the characteristics of the datasets used (low number of dimensions, simple boundaries, etc), even if the combination of SWCNT/LC and implementation is able to solve the classification problems, it is not in itself an indication of whether more complex, real-life problems can be solved using this same process. This is a common problem in unconventional computing, where an ongoing discussion is concerned with the establishment of correct benchmark function to be used evaluating new algorithms or computational framework.

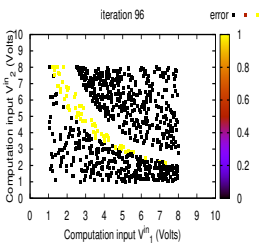
4.9 Summary of Results and Conclusions

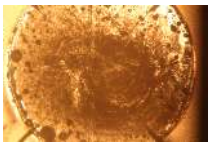
The chapter reported a collection of experiments aiming at evaluating different parameters involved in the EiM implementation. Results from these evaluations support the implementation choices made in subsequent investigations.

Using the EiM process, it was possible to transform a liquid SWCNT/LC composite into a classifier stable enough to solve a binary classification problem (BCP). Under the same implementation, it was not possible to solve a BCP with satisfactory accuracy using bare electrodes or an array of linear resistors. This supported the hypothesis that the SWCNT/LC composite was an integral part of the solution minimising the BCP's classification error. Furthermore, it was observed that the SWCNTs were a necessary addition to the LC, since the BCP could not be solved with LC-only samples.

The choice of material concentration affected solution accuracy and training speed. An optimum concentration range was observed around [0.05, 0.1] wt % SWCNT/LC. This was attributed to the percolation threshold reported for this material [16], and the effect of the concentration on the viscosity of the material, and as a result, on the changes in material morphology produced by the algorithms' search.

Results reported in the chapter suggest that it is possible to evolve SWCNT/LC composites into binary data classifiers using the implementation detailed in Chapter 3. In addition, it is observed that for samples in the [0.05, 0.1] wt % SWCNT/LC range, training efficiency was, on average, better than for other concentrations. This range of concentrations is lower than the [0.7-1] wt % SWCNT/PBMA optimum range observed in [4, 15].





This is consistent with the percolation threshold, observed at lower SWCNT concentrations in SWCNT/LC [16] compared to solid SWCNT/polymer samples [4], and implies that less SWCNTs are required to obtain good solutions with the former composite.

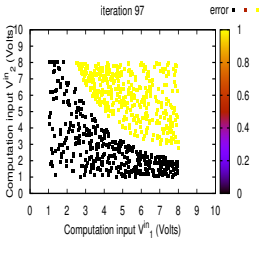
Two algorithms were compared using 0.05 wt % SWCNT/LC samples. A trade-off between result accuracy and consistency was observed. The particle swarm optimisation (PSO) algorithm tended to produce results that converged slowly to a stable solution that generalised well. Solutions obtained with differential evolution (DE) also generalised well, with a classification accuracy that was, on average, higher for three out of four BCP. However, there was more variability in accuracy across experiments when DE was used. It was possible to link these observations with both the search behaviour of each algorithm (focused on exploration or exploitation), illustrated by the configuration voltage trajectories recorded during training, and the resulting material behaviour, illustrated by the rate of change in material morphology during training. The results and discussion extend those previously reported by the author in [22, 23].

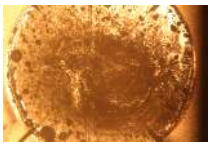
Finally, it was observed that a linear function could be used to translate the SWCNT / LC's outputs into a classification error for the simplest BCPs. For more complex BCPs, improved training speed and result accuracy were obtained by modifying this basic function. However, past a given level of problem complexity, no statistically significant difference in results was observed with further function variations.

Overall, it was observed that although the computation can be considered as analogue in nature, it was the macro-behaviour of the emerging material properties that is used. The alignment and formation of percolation paths of SWCNT within the LC host is enforced by the iterative nature of the evolutionary search conducted, until a notion of computation error is minimised or becomes zero. It must be noted that just as in the analogue computing case, there was a persistent issue with the computation's accuracy. This is a problem which can be addressed by improving the quality of the hardware used and the efficiency of the training algorithm.

Bibliography

[1] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.





[2] M. Dale, J. F. Miller, and S. Stepney, "Reservoir computing as a model for in-materio computing," in *Advances in Unconventional Computing*, pp. 533–571, Springer, 2017.

[3] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[4] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[5] P. Jeatrakul and K. W. Wong, "Comparing the performance of different neural networks for binary classification problems," in *Natural Language Processing, 2009. SNLP'09. Eighth International Symposium on*, pp. 111–115, IEEE, 2009.

[6] M. Srinivas, R. Bharath, P. Rajalakshmi, and C. K. Mohan, "Multi-level classification: A generic classification method for medical datasets," in *2015 17th International Conference on E-health Networking, Application & Services (HealthCom)*, pp. 262–267, IEEE, 2015.

[7] L. Mena and J. A. Gonzales, "Symbolic one-class learning from imbalanced datasets: Application in medical diagnosis," *International Journal on Artificial Intelligence Tools*, vol. 18, no. 02, pp. 273–309, 2009.

[8] M. Lichman, "UCI machine learning repository," 2013.

[9] R. Nayak, P. S. Patheja, and A. Waoo, *An Enhanced Approach for Weather Forecasting Using Neural Network*, pp. 833–839. India: Springer India, 2012.

[10] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

[11] R. A. Fisher, "The use of multpitle measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

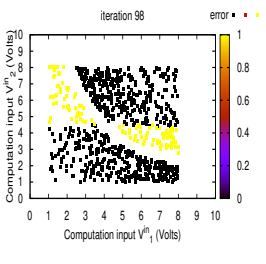
[12] F. Qaiser, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, and M. C. Petty, "Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation," in *IEEE Congress on Evolutionary Computation (CEC)*, pp. 5255–5261, IEEE, 2016.

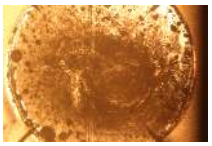
[13] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.

[14] M. Mohid, *Evolution-In-Materio: Solving Computational Problems Using Materials (PhD Thesis)*. University of York, 2015.

[15] M. Dale, *Reservoir Computing In-Materio (PhD Thesis)*. University of York, 2018.

[16] D. Volpati, M. K. Massey, D. Johnson, A. Kotsialos, F. Qaiser, C. Pearson, K. Coleman, G. Tiburzi, D. A. Zeze, and M. C. Petty, "Exploring the alignment of carbon nanotubes dispersed in a liquid crystal matrix using coplanar electrodes," *Journal of Applied Physics*, vol. 117, no. 12, p. 125303, 2015.





[17] M. K. Massey, A. Kotsialos, D. Volpati, E. Vissol-Gaudin, C. Pearson, L. Bowen, B. Obara, D. A. Zeze, C. Groves, and M. C. Petty, "Evolution of electronic circuits using carbon nanotube composites," *Scientific reports*, vol. 6, 2016.

[18] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.

[19] M. Dale, S. Stepney, J. F. Miller, and M. Trefzer, "Reservoir computing in materio: A computational framework for in materio computing," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2178–2185, IEEE, 2017.

[20] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes.," *International Journal of Unconventional Computing*, vol. 10, no. 5-6, pp. 473–497, 2014.

[21] E. Ozcan and C. K. Mohan, "Particle swarm optimization: surfing the waves," in *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406)*, vol. 3, pp. 1939–1944, IEEE, 1999.

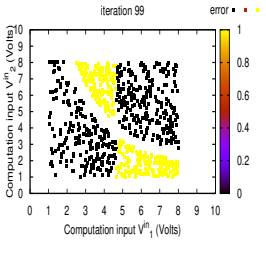
[22] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Training a carbon-nanotube/liquid crystal data classifier using evolutionary algorithms," *Unconventional Computation and Natural Computation Conference: 15th International Conference, UCNC 2016*, pp. 130–141, 2016.

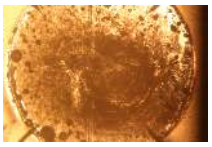
[23] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Data classification using carbon-nanotubes and evolutionary algorithms," *International Conference on Parallel Problem Solving from Nature*, pp. 644–654, 2016.

[24] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[25] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebo, M. K. Massey, and M. C. Petty, "Evolution-in-materio: A frequency classifier using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 46–53, IEEE, 2014.

[26] M. Mohid and J. Miller, "Solving even parity problems using carbon nanotubes," in *Computational Intelligence (UKCI), 15th UK Workshop on. IEEE Press*, 2015.





Chapter 5

Characteristics of Evolved SWCNT/LC Classifying Devices

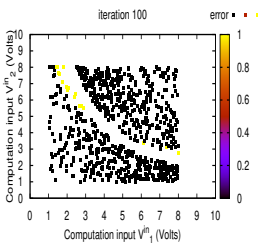
5.1 General Overview

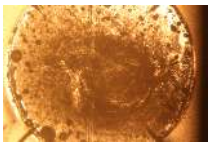
Results and analysis regarding the solving of binary classification problems (BCPs) have been outlined in Chapter 4. The main aim of this Chapter is to investigate whether the use of single-walled-carbon-nanotube (SWCNT)/ liquid crystal (LC) composites can help in understanding the impact of the evolution in materio (EiM) process on materials, and whether there is a benefit to using liquid rather than solid samples. Four sets of investigations are reported, following four research questions:

1. How fast and accurate is the training of the material for evolving binary classifiers, and can the resulting classifier performance be reproduced using different samples of the same composite?

2. What is the contribution of the material itself to the system's behaviour as a binary classifier and what is the contribution of the configuration inputs?

3. Can a successful training programme for one processor be applied to different, nominally similar processors, and get the same result? In other words, can two different samples of the same material, trained using a given sequence of inputs, solve a classification problem with the same or similar accuracy?





4. Can SWCNT/LC samples be reconfigured by the evolutionary algorithm to solve more than one binary classification problem (BCP)? Does the complexity of the problems' dataset affect this process?

A novel approach to analyse the performance of the physical SWCNT-based classifiers resulting from the EiM experiments is also reported in this Chapter. It is observed that a confidence measure associated with the classification accuracy of the evolved material-based classifier can be calculated. The measure is based exclusively on the solution produced by the evolutionary algorithm, ie: a combination of the material's electrical state and a threshold that belongs to the problem's set of decision variables. This is an important contribution in that it allows further investigations to be undertaken with complex datasets where a high confidence in the result is crucial.

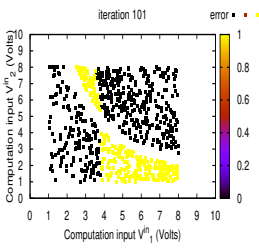
The new measure of classifier confidence for *in materio* computation is introduced and discussed in the first section. In the following four sections, the results of investigations aiming to answer the four research questions are reported and discussed. The final section summarises results and concludes this Chapter.

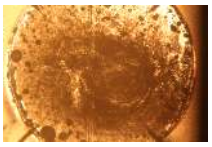
5.2 Confidence Measures in Materio

5.2.1 Motivations

Training errors and verification errors, such as have been described in Chapter 3 and used in Chapter 4, are two common measures of the performance of a classifier produced through supervised learning [1]. They can be used either to compare classifiers obtained through different means, or to estimate how well a given classifier will be performing for a specific task. These measures have the advantage of being generally straight-forward to obtain. In addition, in the case of BCPs where balanced datasets are available, and both classes have equal importance, training and verification errors can provide a good estimate of the error that will be obtained by a classifier when faced with new data. However, these two errors are not always sufficient, and can sometimes even be misleading as to the real quality of the classifier they are used to assess, motivating investigations into additional measures of classifier performance.

A number of classifier performance measures are presented, assessed and compared in [2] and [3]. Overall, both papers argue that the quality of the information provided by



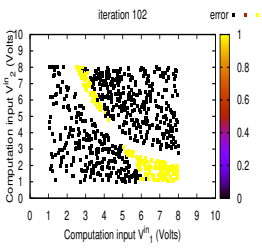


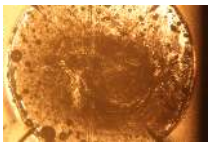
a measure will depend on the problem at hand and the type of classifier used, i.e. there is not one measure which is optimum for all problems/classifiers. Some of these measures are therefore designed with a specific application in mind, such as the recall rate [2], which is generally used when dealing with medical datasets. Others, can be based on statistical analysis, such as the non-parametric statistical tests used to compare classifiers evolved following the reservoir computing *in materio* (RCiM) approach [4]. Other still are classifier-dependent. For example, some measures developed for a probabilistic classifier in [5] are not applicable to an artificial neural network (ANN)-based classifier, for which other measures are proposed in [6, 7].

In the work presented here, the aim is to investigate whether information regarding a classifier's quality can be extracted from measurements collected across evolved devices, that might not be available when using other classifiers, whether it is because they are realised in a different material, or because they have not been obtained using EiM. Firstly, a measure of the reliability of the training errors and verification errors obtained during and post training is reported. It was observed during experiments that training and verification errors did not always match. This lead to the question of whether it is possible to predict by how much these two errors will differ, and how much error can be expected post-verification. In other words, how confident can one be that the training and verification errors are truly representative of the classifier's quality. Secondly, a confidence measure is proposed. Its aim is to provide information about areas of the dataset where instances correctly classified during training or verification tests are most likely to be misclassified in subsequent tests, i.e. areas of high uncertainty.

5.2.2 Experimental Implementation

Three out of the five synthetic binary datasets introduced in Chapter 4 were investigated: V1C, MC and NLC. They represent linearly separable, merged and non-linearly separable classification problems, respectively. The V1C and NLC problems have different complexities associated with the width and shape of their separating boundary, but in both cases, an optimum solution will produce a 0% error. On the other hand, the MC dataset presents an overlapping area between the two classes containing 6.6% of all instances. As mentioned previously, this means that the lowest possible error that can be achieved is 3.3%. The differential evolution (DE) algorithm presented in Chapter 3 is





used to evolve the composites. In each of the $W = 14$ experiments undertaken per problem (except MC, where $W = 4$, as the variation across experiments was ≈ 0), a different SWCNT/LC sample has been used.

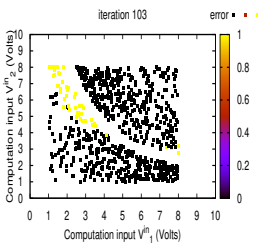
The performance of the evolved classifier is first assessed using the metrics related to the percentage of classification error. In each experiment $j = 1, \ldots, W$, starting from initially untrained material samples, both training and verification are performed, yielding the optimal training error $\Phi_{e,j}^{t,*}$ and the corresponding verification error $\Phi_{e,j}^{v}$, calculated according to eq.(3.22). Based on these experimental results, the following metrics are used,

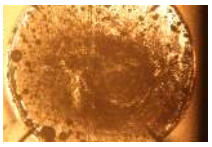
$$\begin{aligned} &\Phi_e^{t,*} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W} \Phi_{e,j}^{t,*}, \quad \overline{d_\Phi} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W} |\Phi_{e,j}^{t,*} - \Phi_{e,j}^{v,*}| \\ &\overline{\Phi_e^v} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W} \Phi_{e,j}^{v} \qquad \text{and } \sigma_{\Phi_e^v}, \end{aligned} \tag{5.1}$$

where $\overline{d_\Phi}$ provides an indication of how well the solution evolved during training (and resulting in $\Phi_e^{t,*}$) generalises to new data, i.e. whether the classifier has overfit to the training data. The average verification over experiments, $\overline{\Phi_e^v}$, in conjunction with the standard deviation $\sigma_{\Phi_e^v}$ evaluating the spread of results across this mean, provide a measure of the results' reproducibility on different material samples. For each metric, 0% indicates optimality, except in the case of the MC problem, where the optimal value of $\Phi_e^{t,*}$ and $\overline{\Phi_e^v}$ is 3.3%.

The figure of merit (FoM) is the name given to the confidence measure developed for the *in materio* classifiers described in this work. When an instance from a dataset is classified, it is assigned a % FoM, or confidence in the class it has been assigned to, based on the physical quantities related to the optimal material state and decision variables, i.e. the solution $\mathbf{x}^*$. Since the class assigned to an instance defined by a set of computation inputs $\mathbf{V}^{\mathcal{C}}$ is determined using an interpretation scheme, as first defined by eq.(3.7), the calculation of the FoM varies according to the choice of classification problem, as discussed in Chapter 4, Section 4.8. In both formulations of the interpretation scheme used for the BCPs, the output currents measured across the material are compared to the best value of the decision variable R_1^*. Based on this comparison, the FoM for the V1C and NLC datasets is given by

$$FoM = \left| \frac{I_1(k)V_1^{\mathcal{C}}(k) + I_2(k)V_2^{\mathcal{C}}(k) - R_1^*}{\max_k \{I_1(k)V_1^{\mathcal{C}}(k) + I_2(k)V_2^{\mathcal{C}}(k) - R_1^*\}} \right| \times 100\ (\%) \tag{5.2}$$





and for the MC dataset it is

$$FoM = \left| \frac{(I_1(k)/I_2(k)) - R_1^*}{\max_k \{(I_1(k)/I_2(k)) - R_1^*\}} \right| \times 100\ (\%). \tag{5.3}$$

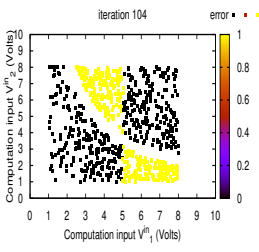
This is effectively a measure of the difference between the output currents collected across the material, when it is sent information about an instance, and the optimum configuration variable R_1^* used in the interpretation scheme. This difference is normalised using the maximum difference achieved in the dataset. The FoM is given as a percentage. Points with 0% FoM can be considered to have been classified at random. A 100% FoM suggests maximum confidence in the class to a point.

5.2.3 Results

During an experiment, the best iteration, λ^*, is the iteration where the best solution, $\mathbf{x}^* = [\mathbf{x}'^*, \mathbf{M}^*]^T$ was produced. The training error, $\Phi_{e,j}^{t,*}$, resulting from the application of the decision variables $\mathbf{x}'^*$ to the material in state $\mathbf{M}^*$, for one experiment j, indicates the quality of the classifier at this iteration, and thus the level of error expected during verification tests. The FoM associated with the training instances was calculated at λ^* for all experiments, in order to determine whether this new measure of classifier performance could provide an estimate of the verification error more accurate than that solely based on the training error.

Figure 5.1(a) presents the FoM obtained for each instance of the V1C training dataset at iteration λ^*, using $\mathbf{x}^*$, and mapped in the 2D computation input space. Figure 5.1(b) presents the distribution of misclassified instances during a verification test where $\mathbf{x}'^*$ was applied to the evolved material along with instances from the V1C verification dataset. For the sake of clarity, the data plotted in these two figures was collected for *one* experiment performed using the DE algorithm to solve the V1C classification problem.

The lighter coloured instances in Figure 5.1(a) have a higher FoM, i.e. the confidence in the fact that they have been assigned to the correct class is high. On the other hand, the darker areas are areas of high uncertainty: FoM values are low and so is the confidence in class assignment. The best training error achieved at λ^* was $\Phi_{e,j}^{t,*} = 0.125\%$, which means that all but one out of the 800 training instances were correctly classified. However, areas where instances have an FoM between 0% and 5% can be observed in the vicinity of the separating boundary between the classes. According to the definition



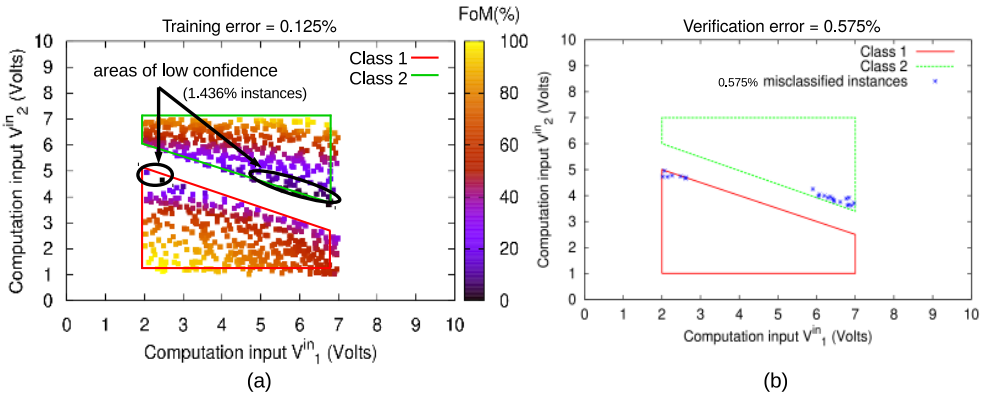
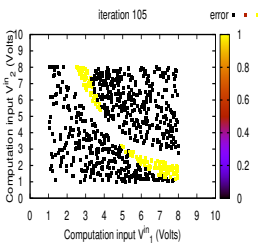


FIGURE 5.1: Mapping of (a) the level of confidence in the class assignment of correctly and incorrectly classified V1C training instances at iteration λ^*, in terms of their FoM values and (b) instances misclassified during a verification test where the solution $\mathbf{x}'^*$ obtained at iteration λ^* was applied to the evolved material and tested against the V1C verification dataset.

of the new measure introduced in this work, instances contained in these darker areas are most likely to have been classified at random.

In this experiment, the training error alone suggested a verification error of 0.125%, which corresponds to 5 misclassified instances from the verification dataset (0.00125×4000). However, 1.436% of the training instances had an FoM within 5%. This suggests that about 57 instances from the verification dataset (0.01436×4000) have the potential to be randomly assigned to the correct or incorrect class, raising the expected verification error to $1.436/2 = 0.718\%$. The 0.125% was not added to the expected error, as the misclassified training instances' FoM was below 5%. The verification error, averaged across the ten verification tests performed post-training for the experiment discussed was $\Phi^{v}_{e,j} = 0.515\%$. For the test presented in Figure 5.1(b), the error was 0.575%. In this case, the FoM provided a better estimate of the verification error as compared to the best training error, $\Phi^{t,*}_{e,j}$ alone. In addition, the location of the misclassified verification instances within classes 1 and 2, as observed in Figure 5.1(b), confirms that the low confidence regions identified during training are those where misclassification occurs.

Figures 5.2(a), (b) present the FoM associated with the training instances of the NLC dataset and the distribution of the misclassified instances obtained during verification. The same data is presented in Figures 5.2(c), (d) for the MC problem. In both cases it can be observed that identifying the areas within the training datasets where instances have an associated FoM $\in [0, 5]\%$, enables the identification of areas with high concentration of misclassified verification instances. This is irrespective of whether all training instances within these areas were correctly or incorrectly classified during training. Similarly, using the FoM value enables the calculation of an estimate of the verification error



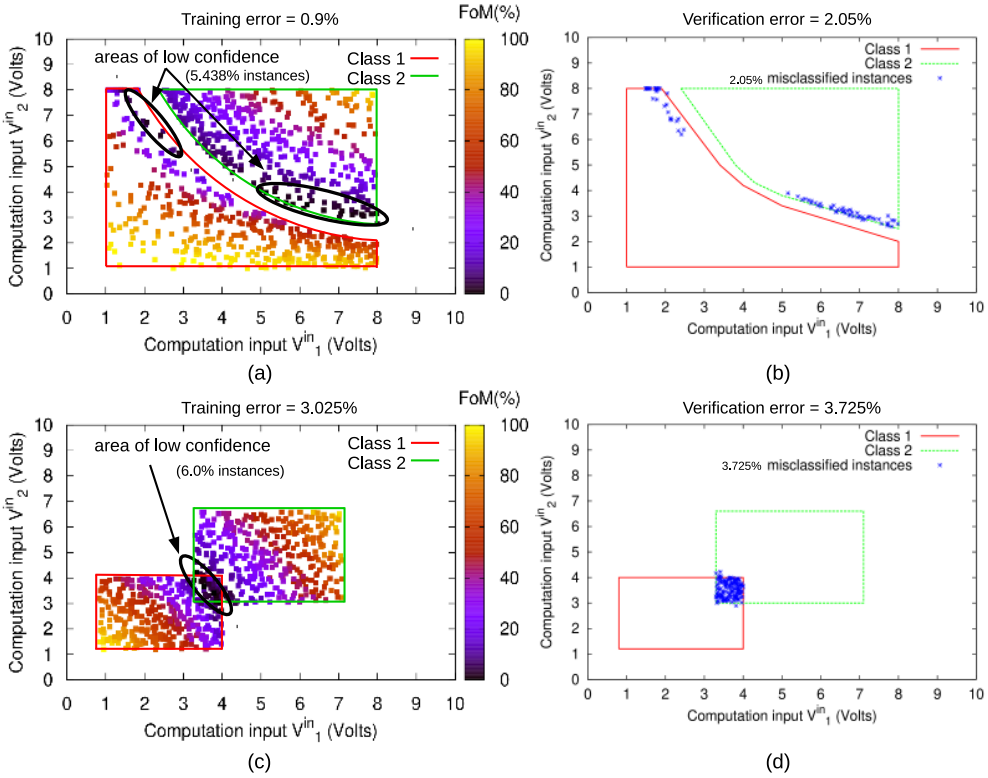
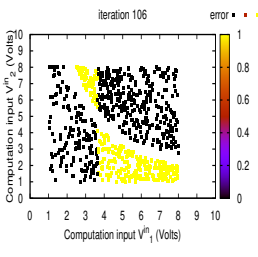


FIGURE 5.2: Mapping of the confidence in the class assignment, in terms of FoM, of the correctly and incorrectly classified training instances at iteration λ^* for the (a) NLC dataset and (c) MC dataset. Distribution of misclassified verification instances for the (b) NLC and (d) MC datasets, respectively.

closer to the actual value of the verification error found post-training, as compared to the estimate provided by the training error.

The optimal training error $\Phi_e^{t,*}$, difference between best training and best verification $\overline{d}_\Phi$, verification error $\overline{\Phi_e^v}$ and standard deviation σ_Φ, are reported in Table 5.1. These metrics are averaged over the ten (and four for MC) experiments undertaken for the three datasets. The FoM values which are also reported in this table correspond to the percentage of correctly and incorrectly classified instances from the training or verification datasets which have an FoM $\in [0, 5]\%$. As discussed in relation to Figure 5.1, these instances have the lowest confidence associated to their class assignment, i.e. they have a 50/50 chance of being assigned to one class or the other. The percentage is therefore divided by two, and the final result can be used as an estimate of the evolved classifiers' ability to deal with new, previously unseen, instances from the relevant dataset.

In the case where incorrectly classified instances had an FoM over 5%, the percentage they represent over the training dataset was added to the final FoM value, without dividing it by two. This choice is justified by the fact that instances with an FoM above



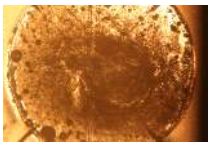


TABLE 5.1: Performance measures for the V1C, MC and NLC datasets.

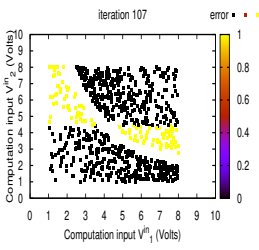
Dataset	$\Phi_e^{t,*}(\%)$	$\overline{d_\Phi}$ (%)	FoM (%)	$\overline{\Phi_e^v}$ (%)	σ_Φ (%)	FoM (%)
V1C	0.121	0.418	0.935	0.694	0.822	0.538
MC	4.831	0.600	5.938	5.648	3.109	17.700
NLC	0.500	4.134	5.103	5.937	8.428	8.745

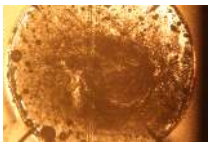
5% are assumed to be always assigned to the same class by the evolved classifier, and therefore if this class is incorrect, always misclassified.

Across the three problems, it can be observed from Table 5.1 that low training errors can be achieved by the algorithm, and that the solution evolved during training can generalise to unseen data. However, the FoM appears to provide a better estimate of the quality of the solution than the training error. In the case of the V1C and MC problems, the FoM gives a potential % of error within 0.3% of the error obtained during verification. This is lower than the estimate provided by the training error, which was up to 1%. In the case of the NLC, the difference is even more pronounced, with the training error suggesting evolved devices able to classify instances with 99.5% accuracy, whilst the FoM suggests an error of 5.103%, which is close to the 5.937% verification error average actually obtained across experiments.

The data plotted in Figure 5.3 (a), (c) and (e) illustrates the percentage of FoM obtained during the verification tests performed post-training, for a single experiment where the verification errors were 0.018%, 3.905% and 0.475% for the V1C, MC and NLC problems, respectively. The cumulated numbers of correctly classified verification instances as a function of the % FoM are represented by full lines. The dotted lines represent the cumulated number of instances that have been incorrectly classified by the evolved material. It can be observed that the datapoints assigned to the wrong class all obtained a very low FoM.

In the case of the V1C dataset, it can be observed that the majority of correctly classified instances had a FoM higher than the misclassified instances. The area of random class assignment lying within 5% FoM contains 0.5375% of all instances from the V1C dataset. The distribution of the correctly and incorrectly classified instances, along with their % FoM is plotted on the 2D map of the dataset within Figure 5.3 (a). In the same figure, an arrow points towards the location on this map where the incorrectly classified

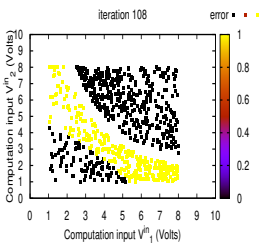


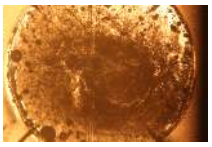


instances can be found. As expected, these areas are the darkest, ie: they have the lowest FoM and therefore the highest probability of being incorrectly classified. Figure 5.3(b) illustrates in three dimensions the % FoM values per instance on the map of the V1C dataset. Instances in Class 1 tend to be closer to the threshold than instances from Class 2. This can be explained by the fact that the voltage inputs that characterise instances from C_2 are higher than those from C_1. The material is in such a state that the outputs are higher for C_2 instances. Multiplying by the computation voltage level further increases this difference. The visualisation of the FoM/instance distribution presented in Figure 5.9(b) also provide an idea of how optimal a processor is: the most optimal processors present the most symmetrical and steepest slopes about the decision line. The average verification error and FoM for V1C across experiments are reported in Table 5.1.

In the case of the MC dataset, the FoM is computed differently, following the interpretation scheme used for this problem. In addition, compared to the other two BCPs which are fully separable, MC presents an area of overlap between the two classes, and its minimum is 3.3% error. This can be observed in Figure 5.3(c), where the misclassified instances are represented by points on the graph. The arrow points towards the area on the dataset's FoM map where instances have the lowest value, and unsurprisingly, this is situated in the overlap. Similarly to the V1C classifier, however, the majority of correctly classified instances from the MC verification dataset have a percentage of FoM higher than 5% FoM, and approximately 80% of the correctly classified instances have a FoM higher than all those misclassified. On average over all experiments for the MC problem, 17.7% of instances have a high probability of being randomly classified, as reported in Table 5.1.

The last two graphs, in Figure 5.3(e) and (f), present the FoM results for the NLC dataset. The verification error is 0.475%, which is close to optimal. The FoM for NLC is calculated in the same way as for the V1C dataset since they share the same interpretation scheme. A noticeable separation in the FoM value between correctly and incorrectly classified instance was observed. Training has produced a material state all misclassified instances have an FoM below 3%, which is less than the 5% considered as the threshold for low confidence in instance classification. In the NLC experiment illustrated in Figure 5.3 (e) and (f), 8.75% correctly classified instances are within the same 5% FoM as the ones assigned to the wrong class. The performance of the evolved device is not as good





compared to the V1C problem, both in terms of verification error and FoM, as reported in Table 5.1, with an average verification error of 5.937% and a high probability for this error to increase to 8.745% if more tests were undertaken, since this is the value of the FoM. It can be observed in the Figure 5.3(e) that the highest probability of error occurring is situated at the classes' boundaries. When mapped on 2D computation input space the threshold value is a hyperbola separating the two classes, and similarly to V1C,

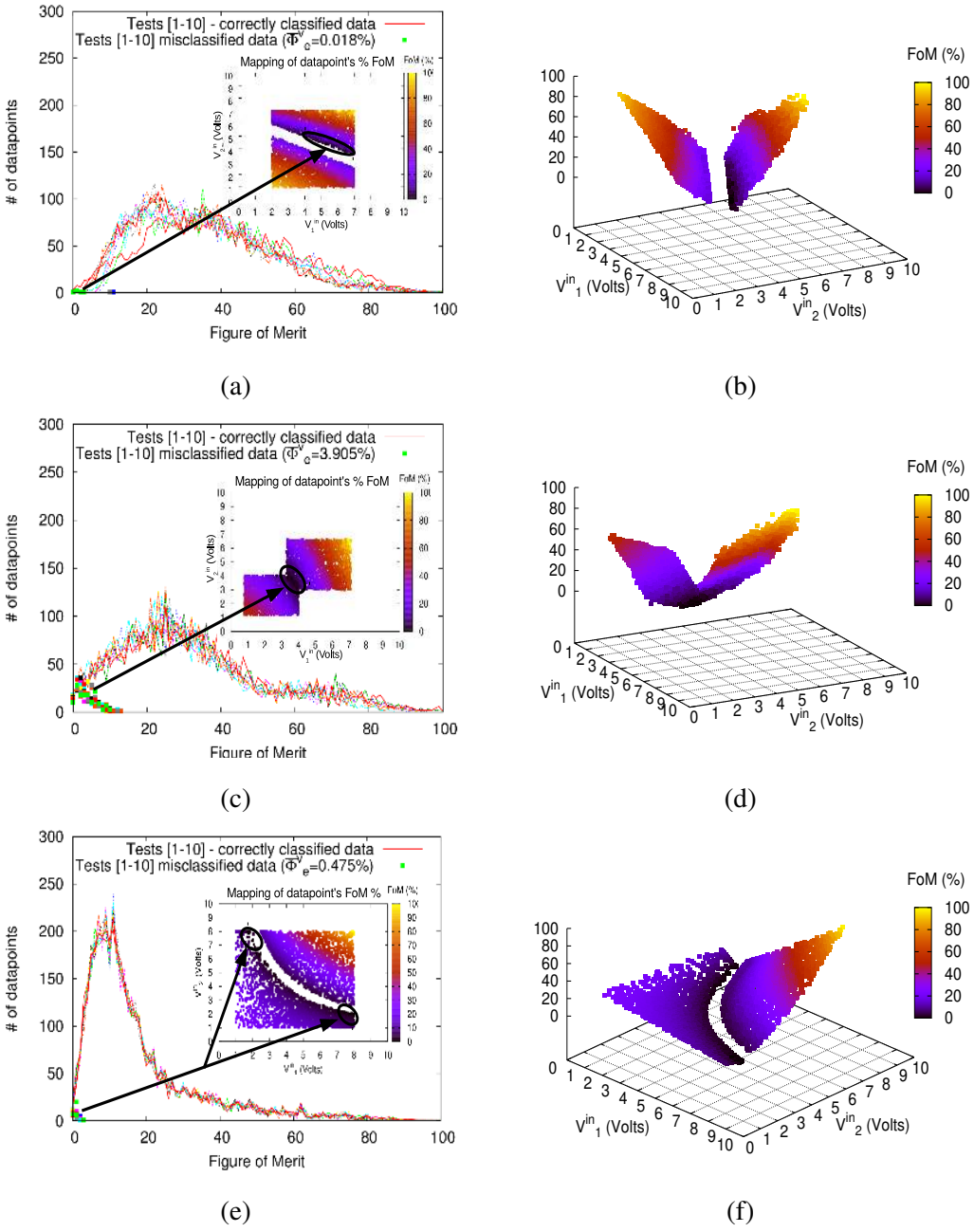
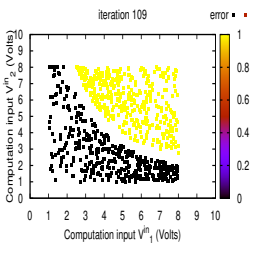


FIGURE 5.3: Distribution of the correctly and incorrectly classified data as a function of their distance from the threshold R (LHS) and mapping of the verification error with associated confidence measure (% FoM) on the computation input space (RHS) for (a),(b) V1C, (c),(d) MC and (e),(f) NLC synthetic binary datasets.



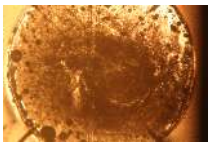


Figure 5.3 (f) shows that a higher confidence tends to be assigned to Class 2.

The measure of confidence was compared with a standard machine learning approach: logistic regression (LR) classifier. Similar to our approach, given an input sample, LR assigns a weight to each possible class and then uses a distance-based measure to perform the classification task. The correlation between the output of the two classifiers is illustrated in Figures 5.4 (a), (b), (c) with 97% Pearson correlation coefficient for V1C, 98% for MC, 92% for NLC, suggesting a high correlation between the proposed classifier in this paper and an established classifier in machine learning.

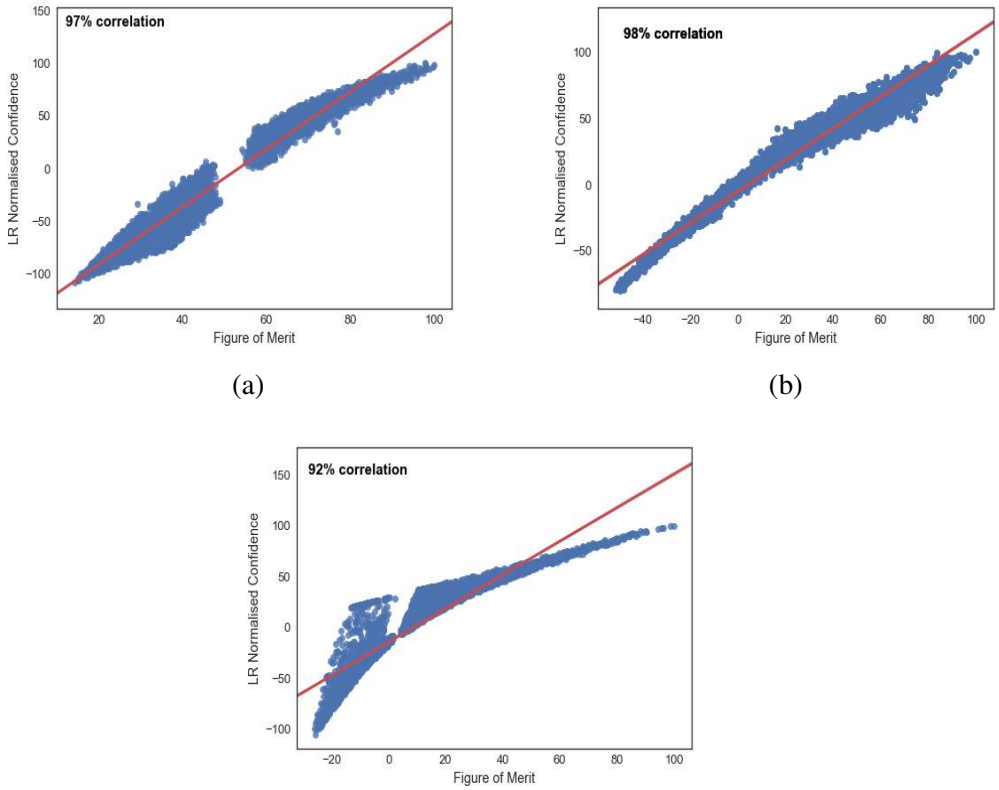
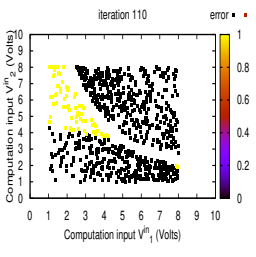
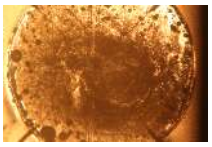


FIGURE 5.4: Pearson Correlation between LR normalised confidence and evolved SWCNT/LC FoM for (a) V1C, (b) MC and (c) NLC

5.3 Efficiency and Reproducibility

Ideally the procedure followed to prepare SWCNT/LC composites aims at creating a uniform dispersion of SWCNTs, and therefore samples with morphological and electrical properties as close to each other as possible. In practice, however, the dispersion is not uniform (Chapter 2) and differences in un-configured sample morphology can be



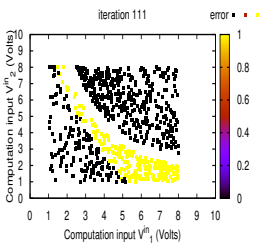


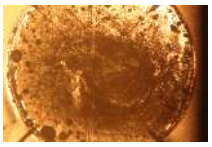
observed at micro-scale. In addition, DE performs a stochastic search, which means that even if every sample had exactly the same properties, the sequence of decision variables produced by the algorithm throughout training could vary from experiment to experiment. The combination of differences between each un-configured samples, i.e. the initial material part of the search space, and the stochastic exploration and exploitation performed by DE raises questions of result reproducibility. Furthermore, the version of DE implemented in Chapter 4 was found to converge to more accurate solutions than PSO, but DE solutions were not always reproducible or able to generalise well to new data instances. This section therefore investigates the reproducibility and efficiency, in terms of accuracy and speed of convergence, of the EiM process implemented with a DE algorithm, as well as the ability of the evolved 0.05 wt % SWCNT/LC processors to generalise to new data.

5.3.1 Experimental Implementation

Investigations consist of a set of $W = 20$ experiments, where a new un-configured sample is subjected to a training and verification procedure. During training, $\mathbf{V}_{\mathbf{t}}^{\mathcal{C}}$ is applied and a sequence of solutions $\mathbf{x}$, is produced by DE until the error in the classification of this dataset is minimised. Verification tests are then performed on the evolved sample, where $\mathbf{V}_{\mathbf{v}}^{\mathcal{C}}$ is used to evaluate the quality of the best solution, $\mathbf{x}^*$, obtained at the end of training. Each verification test is repeated $Q = 10$ times, with a 1s delay between each test. Finally, a new set of ten verification tests is performed with the aim of determining the contribution of the best material state, $\mathbf{M}^*$, to the solution $\mathbf{x}^*$. In this case, all electrodes connected to the material are set to $0V$ for $t = 300$ seconds after the first set of verification tests. Then, using $\mathbf{R}^*$, the verification dataset is reapplied to the evolved device, but the electrode where the optimum set of configuration voltages are applied remain set to $0V$.

In order to analyse results obtained in the series of experiments described, the metrics previously discussed are used. In each of the twenty experiment, both training and verification are performed, yielding the optimal training error $\Phi_{e,j}^{t,*}$ and the corresponding





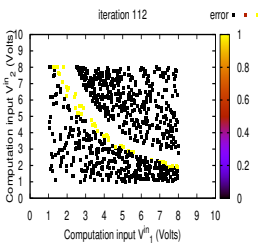
verification error $\Phi^{v}_{e,j}$. The metrics for $j = 1, ..., W$ are

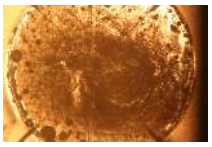
$$\begin{aligned} \Phi^{t,*}_{e} &= \tfrac{1}{W}\textstyle\sum_{j=1}^{W}\Phi^{t,*}_{e,j}, \quad \overline{d_{\Phi}} = \tfrac{1}{W}\textstyle\sum_{j=1}^{W}|\Phi^{t,*}_{e,j} - \Phi^{v,*}_{e,j}| \\ \overline{\Phi^{v}_{e}} &= \tfrac{1}{W}\textstyle\sum_{j=1}^{W}\Phi^{v}_{e,j} \qquad \text{and } \sigma_{\Phi^{v}_{e}}, \end{aligned} \tag{5.4}$$

where $\Phi^{t,*}_{e}$ indicates the samples' best training error averaged over all experiments, which is optimal at 0% for the fully separable datasets, SC, V1C and NLC and 3.3% for the merged MC dataset. Combining $\Phi^{t,*}_{e}$ and λ^{*}, the number of iterations required to obtain the minimum training error, provides a measure of efficiency for the EiM process implemented. The process is considered efficient when λ^{*} and $\Phi^{t,*}_{e}$ are minimal. The difference between best training and best verification errors, $\overline{d_{\Phi}}$, averaged over experiments, indicates the capacity of the solution evolved in SWCNT/LC samples to generalise to unseen data. The average verification error over experiments, $\overline{\Phi^{v}_{e}}$, in conjunction with the standard deviation $\sigma_{\Phi^{v}_{e}}$, provide a measure of the results' reproducibility on different samples. Finally, $\overline{\Phi^{v}_{e}}, \mathbf{V} = \mathbf{0V}$ is the error obtained when no configuration voltages are applied, averaged over all experiments. If the EiM process, in modifying the SWCNT network [8] has created a material state which contributes, or is sufficient for the device to classify data, the error obtained when no configuration voltages are applied to the evolved sample should be the same, or similar to the verification error, $\overline{\Phi^{v}_{e}}$.

5.3.2 Results

A set of preliminary experiments were first undertaken with a number of different algorithm and implementation parameters for DE. The results obtained with each different implementation are not reported here, but it was observed that increasing the number of individuals within the DE population from $N = 8$ to $N = 10$ improved solution reproducibility and generality, without loss of classification accuracy or speed of convergence. This can be seen as surprising, since, contrary to PSO, increasing the population size of DE does not automatically yield better results (cf. Appendix B). In addition, introducing more individuals in the search will induce more changes in the material between iterations. However, the problem of training a material to perform a computation is very different from the test functions used to compared implementation parameters of DE in [9], where it is observed that optimal parameters for this algorithm are, in any case, problem dependent. The results reported in the following experiments were always obtained





with this new DE implementation.

Following the procedure described in the previous section, the control samples - LCs and the array of linear resistors - were tested, and training and verification errors remained around 50%. A similar value was obtained by the un-configured SWCNT/LC composite at the start of an experiment. In the latter case, however, the evolutionary algorithm was subsequently able to minimise this error throughout a number of iterations. Four representative examples of the convergence of the objective function for the four datasets, from 50% to the optimal, are illustrated in Figure 5.5(a). For each example, one out of the eight configuration voltages that modified the material, producing the objective functions, is presented in the Figure 5.5(b). It can be seen that the average voltage trajectories, i.e. sequence of $\mathbf{V}$, produced by the algorithm during training, are different across the datasets. This is also the case across the twenty samples trained to classify instances from the same dataset, but where the original material state $\mathbf{M}$ was different. Figure 5.5(c) and (d) relate to the variable p, which is part of $\mathbf{R}$. It can be noticed that at the point where the value of p becomes constant, the objective function

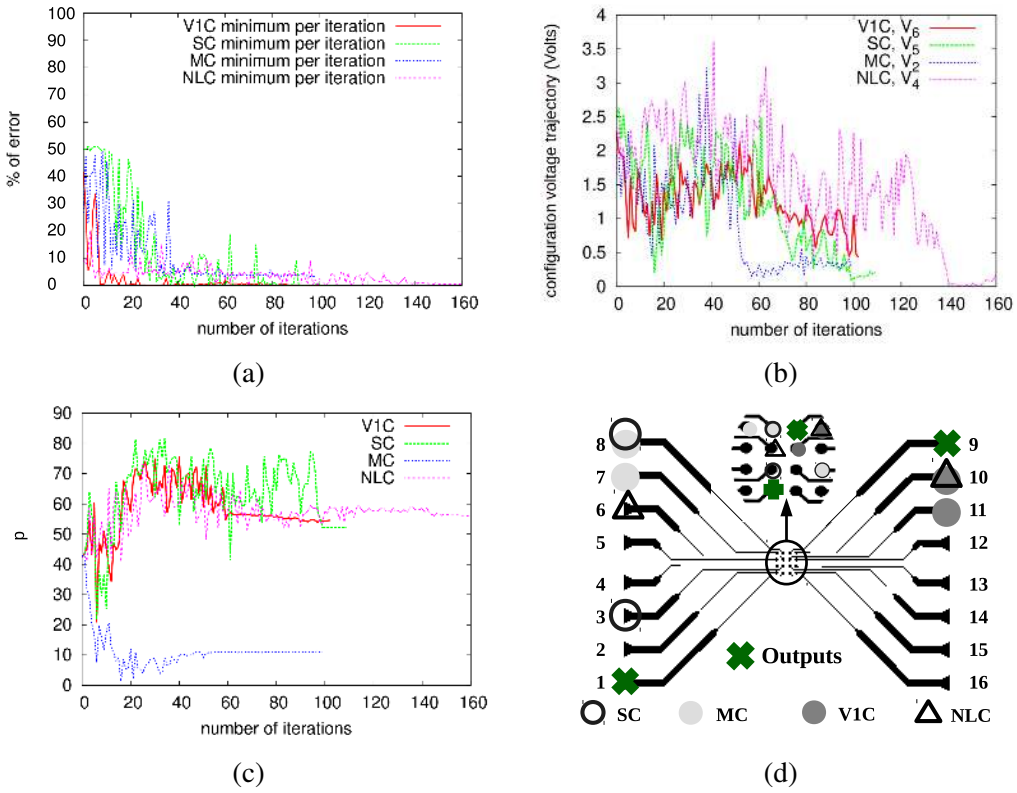
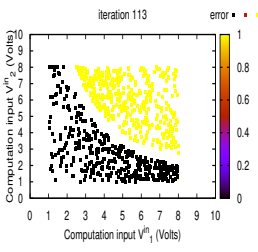
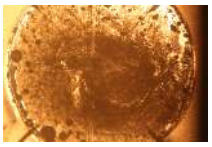


FIGURE 5.5: For four samples, each trained to solve one of the four artificial binary datasets (a) convergence of the average and minimum error (b) representative configuration voltage trajectories (c) evolution of the p, and (d) most common representation of p on the electrode array





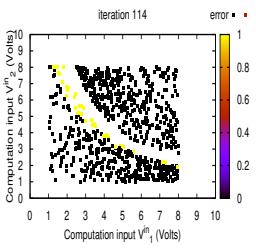
converges rapidly to an optimum. Over the four datasets, this observation holds for the majority of experiments, specifically those where the best results were achieved. It reinforces the suggestion [10] that p is an important parameter of the search process. The resulting most common electrodes, over all experiments, chosen by the algorithm as part of the optimal solution are presented in Figure 5.5(d). No direct correlation between the location where input voltages are sent and a dataset's complexity can be drawn from experimental results.

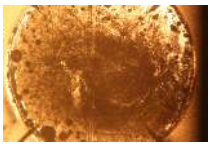
Results obtained with the SC, V1C, MC and NLC dataset are reported in Table 5.2. On average over twenty experiments, and for the four problems, DE was able to consistently converge towards a solution that generalises well, in a relatively low number of iterations. The best training errors were, on average, below 1.5% of each problem's optimum, and were obtained within 215 iterations. In other words, it was possible to create SWCNT/LC processors capable of correctly classifying more than 99% of instances from three separable datasets of increasing complexity within approximately three hours. These processors were then able to generalise to unseen data with less than 3.0% error increase (maximum for the NLC problem, with $\overline{d_\Phi} = 2.737\%$) between training and verification. A difference between training and verification errors is common in classifiers trained using supervised learning, although the aim is to minimise it. In the case of SWCNT/PBMA samples trained using PSO, an average difference of 6% was reported in [11] for datasets similar to the SC dataset. Here, however, the difference can also be the result of a drift effect on the optimum material state $\mathbf{M}^*$, as previously mentioned in Chapter 3, Section 3.2.3.

For the V1C and NLC problems, and across all experiments, a strong (0.7075) and a moderate (0.526) correlation was found between the amount of change in the material state induced by training and the difference in error between training and verification bests, d_Φ, respectively. Since the material state is affected by the voltages applied to

TABLE 5.2: Reproducibility of experiments over SWCNT/LC samples.

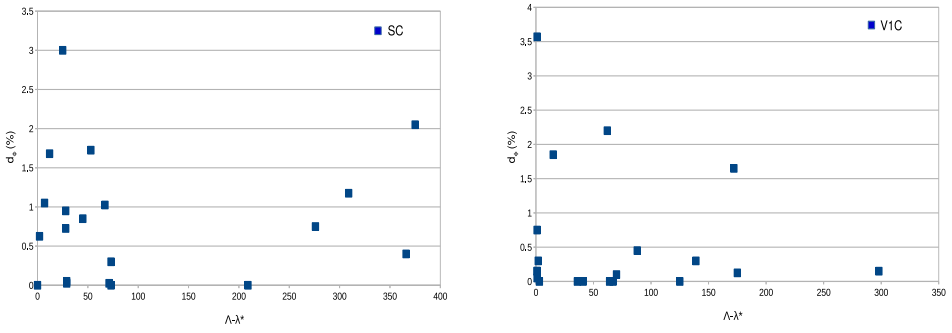
Datasets	$\Phi_e^{t,*}(\%)$	λ^*	$\overline{d_\Phi}(\%)$	$\overline{\Phi_e^v}(\%)$	$\sigma(\%)$	$\overline{\Phi_e^v}(\%), \mathbf{V} = 0V$
SC	0.255	96.85	0.820	1.534	1.522	47.738
V1C	0.035	73.90	1.096	1.525	1.807	**1.178**
MC	5.243	168.6	0.493	6.024	3.379	44.815
NLC	0.710	214.95	2.737	4.721	4.743	**5.193**





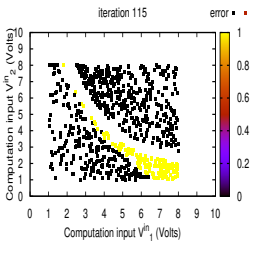
it, a large number of iterations has the potential of creating more important perturbations in the material than a small number of iterations. The change in material state was therefore estimated as the number of iterations between the best iteration, λ^*, and the last iteration before the end of training, Λ. Both correlations were significant at the 0.05 confidence level. In other words, for these two problems, if $\Phi_e^{t,*}$ was found at the beginning of the training process, $\overline{\Phi_e^v}$ was more likely to vary from $\Phi_e^{t,*}$ and across verification tests. This suggests on the one hand that the optimum material state, $\mathbf{M}^*$, had a non-negligible contribution to the overall best solution $\mathbf{x}^*$, and on the other hand, that performing more function evaluations after the optimum was reached induced changes in the material likely to affect the quality of the evolved classifier. In the case of the SC and MC problems, the correlations were weak (< 0.3), i.e. the number of iterations between λ^* and the end of training did not affect the level of generalisation of the solution over the experiments performed here. Figure 5.6 (a) and (b) illustrate the difference in correlation between results obtained with the SC and V1C problems, respectively. The difference between best training and best verification errors are plotted as a function of the difference between the best iteration and the last iteration.

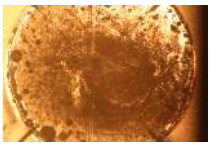
The average verification error and standard deviation results reported in Table 5.2 support the suggestion that the DE algorithm can consistently converge towards good solutions, i.e. solutions that minimise the classification error. As the metrics previously discussed, $\Phi_e^{e,*}$, λ^* and $\overline{d_\Phi}$, the verification error and the amount by which this error varies across experiments are problem dependent, and more specifically appear to increase with increasing problem complexity. However, this is not the case for the results



(a) (b)

FIGURE 5.6: Number of iterations between the best solution resulting in the lowest % training error was obtained, and the end of training ($\Lambda - \lambda^*$), as a function of the difference in error between training and verification bests (d_Φ) for the (a) SC problem and (b) V1C problem.





obtained in the second set of verification tests and reported in the last column of Table 5.2. In this case, no configuration voltages were applied, but $\mathbf{R}^*$ was used to determine the location where the computation voltages should be sent. It can be observed that for MC and SC datasets, the verification error increases from near optimal to $\overline{\Phi_e^v} \simeq 46\%$ when $\mathbf{V} = 0V$. The optimum configuration voltages, however, were specific to the devices. If $\mathbf{V}^*$ and $\mathbf{R}^*$ were sent to other samples (thus different $\mathbf{M}^*$), $\overline{\Phi_e^v}$ remained around 50%. Results obtained over twenty experiments suggest that both the configuration voltages and evolved material state produced by DE training were necessary to classify instances from the MC and SC datasets accurately. This is also consistent with correlation results.

In the case of the V1C and NLC datasets, there is no photographic evidence that the SWCNT structures have been retained after the first verification test. Instead, verification errors are used to quantify the stability of these structures. Without applying the best set of configuration voltages $\mathbf{V}^*$, samples brought into a state $\mathbf{M}^*$ are able to classify unseen instances with an error of 0.847% and 4.45% for V1C and NLC, respectively. For V1C, the error was 0.355% lower than when $\mathbf{V}^*$ was sent to the trained sample. For NLC, this error is 1.48% higher as compared to the event where the full solution, $\mathbf{x}^*$, was used. For the four problems, SC, V1C, MC and NLC, Figure 5.7 illustrates the range of errors obtained when the full solution is applied to the evolved material ($\mathbf{x}'^*$) and when the evolved samples are tested against the verification dataset, but all configuration voltages are set at 0V ($V = 0$ Volts). Irrespective of outliers, a very large difference between the

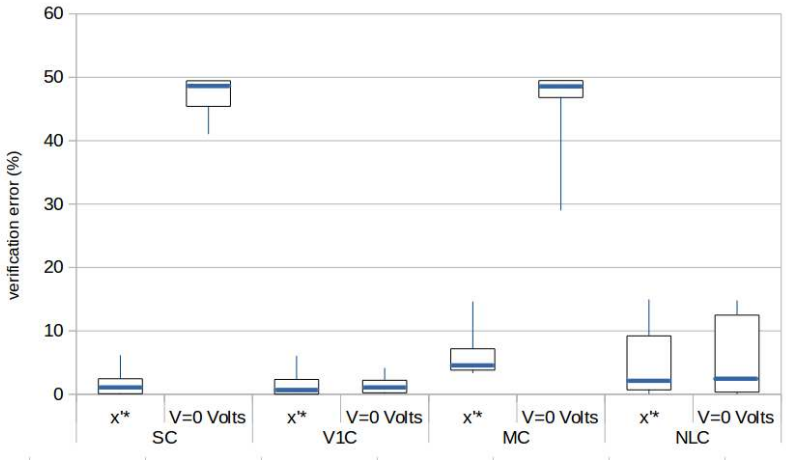
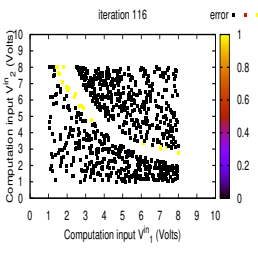
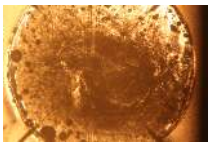


FIGURE 5.7: SC, V1C, MC and NLC verification errors obtained with and without the optimum set of configuration voltages applied to the evolved device in terms of minimum and maximum values, inter-quartile range and median.





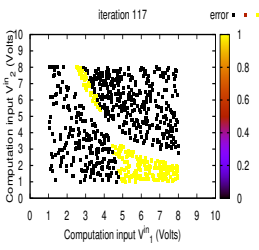
level of verification error with and without applied configuration voltages is observed for the SC and MC problems. On the other hand, the level and distribution of error remains relatively similar in these two cases for the V1C and NLC problems, supporting the discussion based on the results reported in Table 5.2. From the differences in verification errors with and without applied configuration voltages observed with V1C and NLC, it can be suggested that the contribution of the evolved material state is not negligible and can even yield better results than with the configuration voltages. Further, the percolation paths have not relaxed to the original un-configured condition.

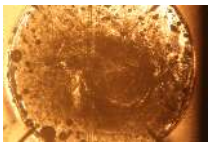
It is proposed that results are due to the fact that for simple problems, the algorithm quickly finds an optimal solution, based almost entirely on the electric field created by the vector of decision variables, without having to modify greatly the materials' morphology. $\mathbf{M}^*$ remains very similar to the pre-trained state of the SWCNT/LC sample, which would explain why verification error is very high when the configuration voltages are set to 0V, On the other hand, with more complex problems where the solution is harder to reach, DE will have to explore and exploit more of the material, forming it, iteration by iteration, into a processing nanotube circuit.

In summary, the process of evolving SWCNT/LC samples into classifiers, using DE, was reproducible for the four artificial datasets of increasing complexity. Despite variations in the structure of SWCNT networks across un-configured samples, using the solution $\mathbf{x}^*$ at the end of training resulted in mean verification errors within 1% of the optimum for the linear merged and the linear separable datasets, and 3.85% for the more complex NLC. For all datasets, the dispersion of results across experiments defined by the standard deviation σ was lower than 2.62%. The contribution of the evolved material state $\mathbf{M}^*$ to the solution was found to be non-negligible for the simplest datasets and sufficient in itself for the more complex NLC and V1C.

5.4 Training Automation (Material Programming)

It was observed from Figure 5.5(b) that the sequence of configuration voltages $\mathbf{V}$ varied across the different datasets. Irrespective of the dataset, this is also the case when comparing the sequences of $\mathbf{V}$ produced by DE when evolving multiple SWCNT/LC samples to classify instances from the same dataset. This observation can be extended to the sequences of $\mathbf{R}$, and, according to some visual and electrical evidence, $\mathbf{M}$. This





phenomenon can be explained by the fact that the search spaces explored and exploited by the evolutionary algorithm were different, thus resulting in the different sequences.

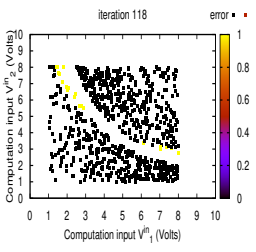
But what would happen if the trajectories of each individual obtained by the optimisation algorithm, when training a particular sample, were to be re-applied on a new initially untrained one, with similar composition? Successful training of new samples would indicate the potential of repeating the process without the computational overhead of running the optimisation algorithm. If so, the complete sequence $\mathbf{V}_\ell(\lambda)$ could be considered as a set of instructions given to the material forcing the creation of internal structures leading to the solution of the particular problem in conjunction with $\mathbf{R}^*$, which does not need to be recomputed. The research question is effectively whether an automation of the transformation of amorphous SWCNT/LC samples into classification devices is possible.

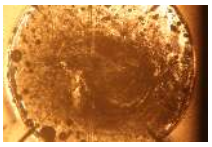
5.4.1 Experimental Procedure and Implementation

In order to investigate the possibility of re-using evolved training sequences, i.e. creating a program of material training, the following procedure was implemented. A sample D_1 was trained from an un-configured state using the DE algorithm. The best solution obtained during training, combined with the evolved material state, was tested against the verification dataset. The convergence trajectory $\mathbf{x}'_\ell(\lambda)$ for $\lambda = 1, \ldots, \lambda^*$ and $\ell = 1, \ldots, N$ produced during the training of sample D_1 was recorded and referred to as $Seq(1)$. Subsequently, three different samples, D_2, D_3 and D_4 of the same 0.05 wt % SWCNT/LC composite were subjected to the configuration voltages $\mathbf{V}_\ell(\lambda)$, in the same exact sequence as they were calculated and applied during D_1's training, without undertaking the computation steps of the optimisation algorithm. The resulting evolved material states of samples $D_2 - D_4$ were assessed by calculating the training and verification errors after iteration λ^* using $\mathbf{R}_{\ell^*}(\lambda^*)$ of the starting point of training.

5.4.2 Results

Training sequences produced by the DE algorithm to solve the V1C and NLC problems have the potential for transferability, since it was suggested in Section 5.3 that they tend to modify the material state such that the solution is a combination of optimum decision variables and the evolved material state, the latter forming a non-negligible part of





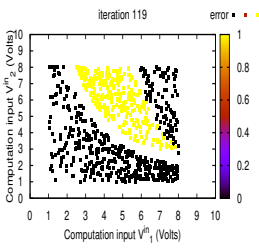
this solution. Compared to the SC and MC problem, it is therefore more likely that the training sequences produced to solve the V1C and NLC problems on given SWCNT/LC samples can be re-applied to new un-configured samples and yield similar results. The results obtained with the V1C and NLC datasets, following the procedure described in Section 5.4.1 are therefore reported in Table 5.3. The training and verification errors for samples $D_1 - D_4$ are presented, along with the standard deviation over the ten verification tests undertaken at the end of training. The latter value indicates the stability of the evolved solution for each sample.

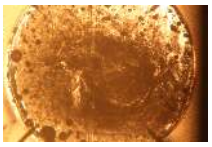
The training of D_1 resulted to good solutions for V1C, where the training error is zero and the verification error 0.649%. The quality of the solution for NLC is not the same as for V1C, since from a zero training error the verification error becomes 9.21%. Nevertheless, the outcome of the application of D_1's training sequence, $Seq(1)$, to the other three samples shows that it is transferable and that an untrained material can be evolved into a classifier performing with similar accuracy. Subsequent application of the optimum solution to the NLC problem obtained for D_1, $\mathbf{R}^*$ and $\mathbf{V}^*$ to the samples $D_2 - D_4$ results in a verification error that is on average 0.52% higher than that obtained with the evolved D_1 samples. Whilst the error levels themselves are higher than those obtained with the $Seq(1)$ produced to solve the V1C, in the case of the NLC problem the difference between D_1 and $D_2 - D_4$ evolved using the same training sequence is lower.

Figure 5.8 illustrates the process where two samples, D_1 and D_2, are trained using the same sequence of configuration voltages, produced by the DE algorithm when looking for a solution to the V1C problem with D_1. The first graph, on the bottom left hand side of the figure presents the convergence of the objective function, or training error,

TABLE 5.3: Transferability of a training sequence on three new samples.

Material Sample	V1C $\Phi^{t,*}_{e,j}(\%)$	V1C $\Phi^{v}_{e,j}(\%)$	V1C $\sigma(\%)$	NLC $\Phi^{t,*}_{e,j}(\%)$	NLC $\Phi^{v}_{e,j}(\%)$	NLC $\sigma(\%)$
D_1	0.00	0.649	0.98	0.00	9.21	3.632
D_2	0.00	2.040	1.780	0.00	10.225	2.957
D_3	0.00	1.070	1.250	0.00	9.945	2.514
D_4	0.00	5.400	2.580	0.00	9.143	2.620





obtained during training for the two samples. The second graph, on the bottom right hand side of the figure, presents the output currents measured across the two samples during training. The training error is a function of the outputs, however, the interpretation scheme defined in eqs. (4.4) and (4.12), and used to translate current outputs into this error, reduces the differences in outputs between the two samples. This results in two evolved SWCNT/LC processors with similar classifier performances. The state $\mathbf{M}$ of the SWCNT/LC sample is evaluated in terms of its electrical response under the same voltage inputs (the output currents measured across the sample in the bottom right plot of Figure 5.8), along with the morphology of the SWCNT structures, as observed from the surface of the sample using a microscope.

The images reported in the top left and top right corners of Figure 5.8 were taken at iteration $\lambda = 59$, which is the last iteration reported in the two graphs reproduced at the bottom of the picture. It can be observed that the morphology of the two materials is different, despite the fact that they have been subjected to the same training sequence, and

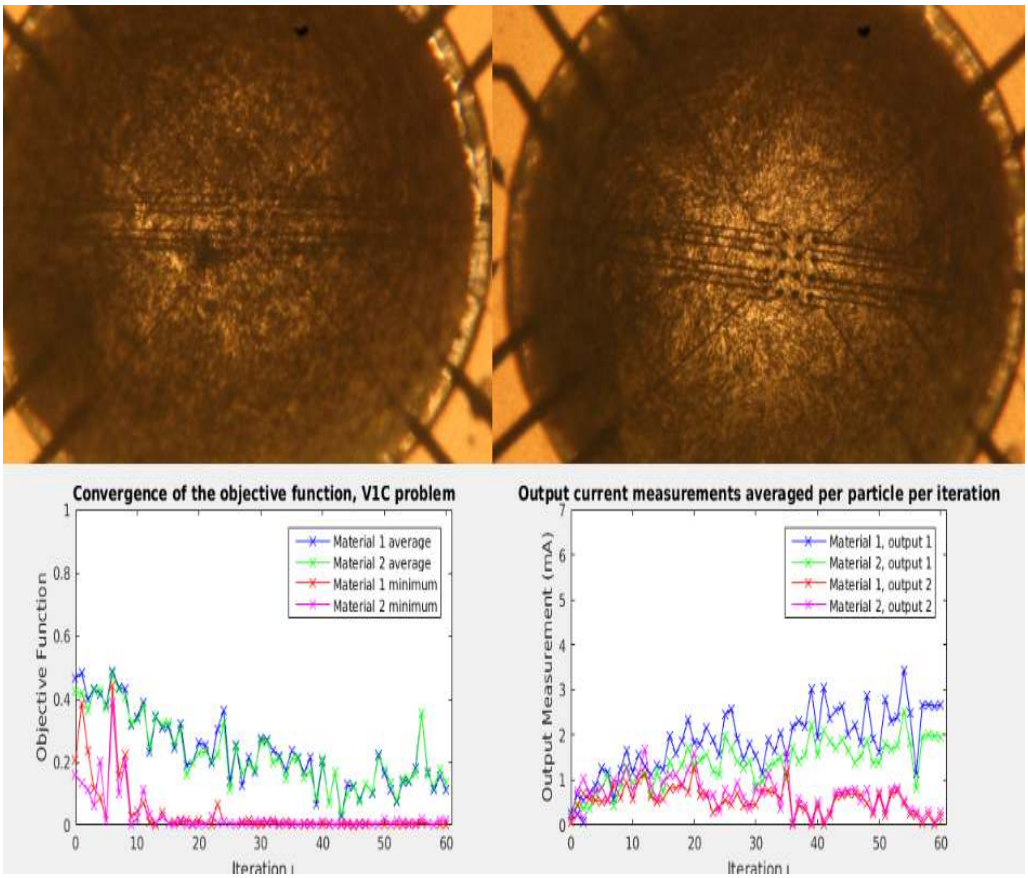
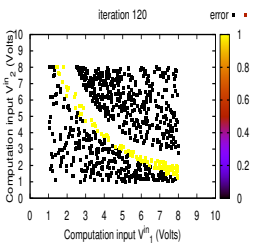
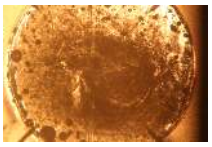


FIGURE 5.8: Final material state of the evolved SWCNT/LC D_1 (LHS) and D_2 (RHS) samples at the top, along with the average convergence of the training error (LHS) which is a function of the current outputs collected across the two samples throughout iterations (RHS) at the bottom.



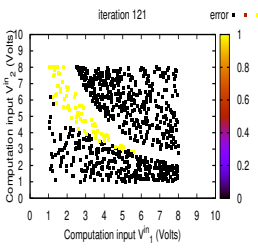


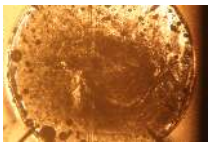
have resulted in classifiers with similar classification accuracy. Table 5.3 and Fig. 5.8 suggest that a previously successful sequence of configuration voltages can be used to evolve different un-configured SWCNT/LC samples with low degradation in classification accuracy. A reduction of the training time by approximately 50% is the benefit of using such a sequence.

When applying $Seq(1)$ to different samples, it is possible to create devices able to classify data with the same accuracy as D_1. Thus a single sequence of configuration variables can produce a number of devices with low variation in accuracy across trained samples. A run of the DE algorithm to evolve a sample D_1 into a data classifier produces $Seq(1)$, which can be considered as a 'program'. This 'program' can be re-used over different samples, despite the inherent variations in the SWCNT networks present in the LC matrix at the start of each experiment. On average, however, the verification error of samples for which DE produced the training sequence is lower than for those upon which the $Seq(1)$ is subsequently applied. There is a trade-off between time and accuracy. Automation of the process makes it faster, i.e. it reductions $\sim 50\%$ of the training time that was due to collection of current outputs and the evaluation of the objective function). However, comparing Tables 5.2 and 5.3 it can be observed that resulting devices lose up to 1.633% accuracy as compared to those trained by DE directly.

5.5 Reconfigurability

Optimisation problem (3.12)–(3.15) refers to a single BCP and the material is trained for addressing a single problem each time. However, it can be desirable to be able to train the one sample to address multiple problems. Since training has been observed to modify the morphology of SWCNT/LC samples, the state of the material at the start of a subsequent training on the same sample will differ from that of the original un-configured sample, as drop-cast at the start of an experiment. This has been generally considered an issue of dynamic samples [12], and EiM investigations have focused on materials with a reset property. Here, investigations focus on the impact of material memory on the quality of solutions found by an EA in the SWCNT/LC samples. It is possible that multiple training is able to destroy structures evolved in a sample during training, and produce new structures favouring the solving of a new problem. It is also possible that the evolved structures with a non-negligible contribution to the solution for





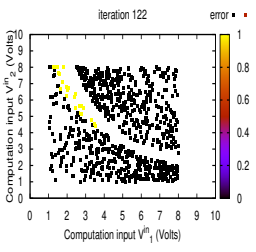
a given problem help the solving of a subsequent one, thereby presenting an advantage over materials with re-set properties. These possibilities are investigated here.

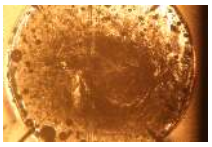
5.5.1 Experimental Procedure and Implementation

The experiment consists of a set of two consecutive training and verification procedures, using two different datasets at a time, and one sample. The un-configured sample is drop-cast onto the micro-electrode array and subjected to a varying electric field produced by the DE algorithm until it is able to solve the first problem. Following this first training, the evolved sample is sent the corresponding verification data along with the optimum set of decision variables, referred to as $\mathbf{V}_1^*$ and $\mathbf{R}_1^*$ obtained during this training. This verifies that the solution $\mathbf{x}_1^*$, combining $\mathbf{V}_1^*, \mathbf{R}_1^*$ and material state $\mathbf{M}_1^*$ can generalise. The second verification test is then performed after 300 seconds during which the configuration voltages are not applied. As seen in Chapter 4, the SC, V1C, MC and NLC datasets present different complexity, but can be relatively similar in structure. Verification data for the second problem is therefore sent along with $\mathbf{V}_1^*$ and $\mathbf{R}_1^*$ on the once trained material to determine if the solution is common to both problems used, and if any further training would be redundant. If it is not the case, the SWCNT/LC material is subjected to a second set of training and verification for the new dataset. The second solution $\mathbf{x}_2^*$ is tested against the verification instances of the second dataset, and the latter are then re-sent with no configuration voltages. Finally, the ability of the doubly trained material to solve the original problem, with and without configuration voltages is assessed.

5.5.2 Results

A graph outlining the experimental procedure and including the distribution of the misclassified verification data for a sequence of tests is presented in Figure 5.9. In this case, a sample was first trained to solve the V1C problem. The ability of the evolved sample to solve the NLC dataset was then tested. Since it was not able to solve it with sufficient accuracy, the sample was retrained using DE to solve the NLC dataset. At the end of the second training and verification procedure, the ability of the doubly-trained sample to solve the original problem, i.e. V1C, was tested. Results obtained for the V1C-NLC experiments illustrated in Figure 5.9 are presented in Table 5.4, along with those obtained for the SC-MC, MC-SC, SC-V1C and NLC-V1C experiments. Values reported in this





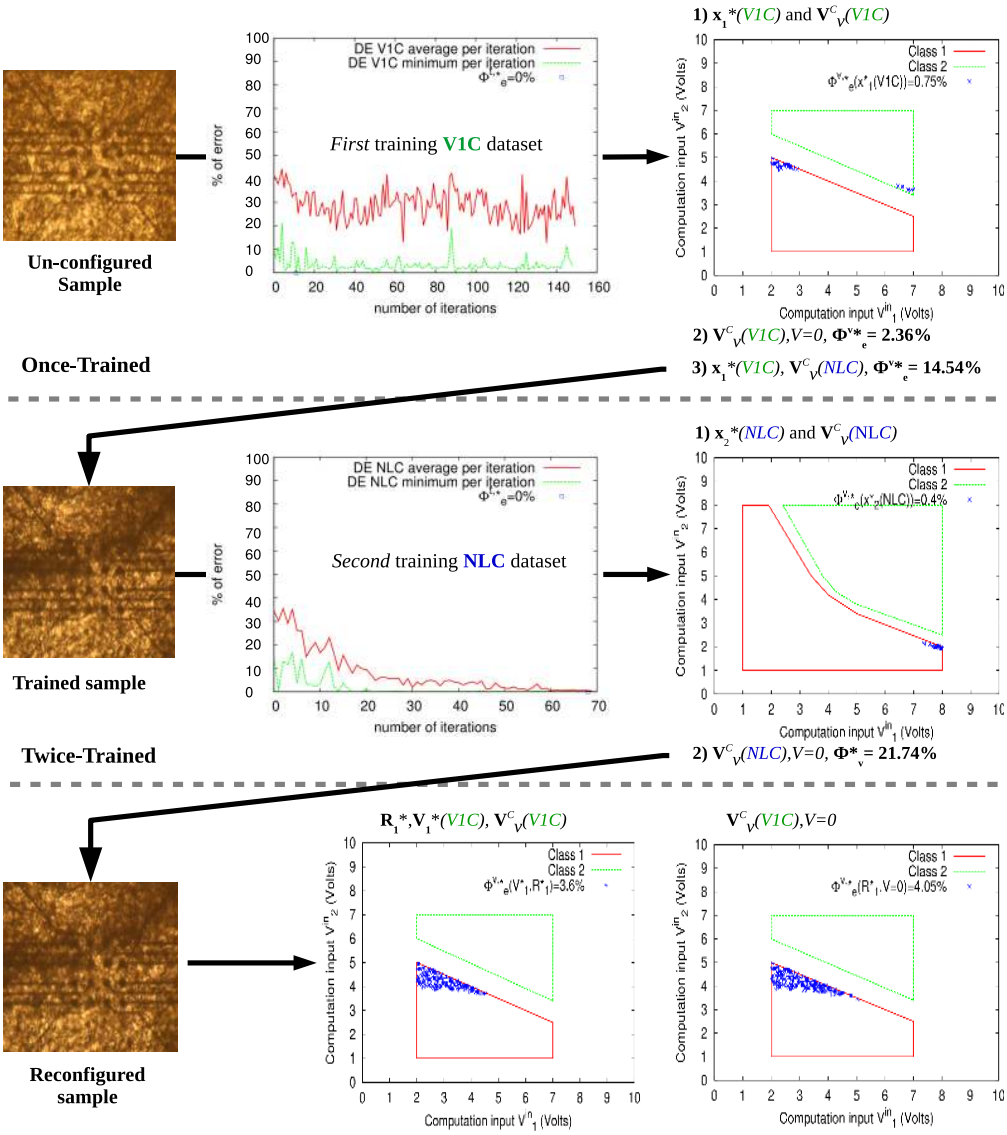
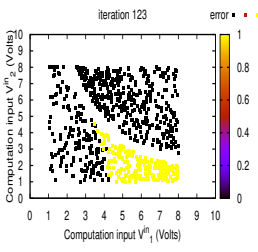


FIGURE 5.9: Simple schematic representation of the reconfiguration process. Results are obtained for one V1C-NLC experiment. First training with the V1C dataset, and second with NLC. The doubly-trained material is tested against unseen instances from both datasets.

table are averaged over three experiments, which was considered sufficient in sight of the reproducibility of the EiM process.

The first results reported in row 1 of Table 5.4 were obtained during the SC-MC experiment. In this case, where the simplest dataset (SC) is used first, the DE algorithm is able to bring the SWCNT/LC mixture into a state where it is able to correctly assign 100% of the *SC* training instances. Testing the solution $\mathbf{x}_1^*$(SC) against unseen data results in a verification error of 1.29%. With no signals sent other than $\mathbf{V}_\mathbf{v}^\mathcal{C}$(SC), the error becomes 48.87%, which is consistent with previous results. The solution $\mathbf{x}_1^*$(SC)



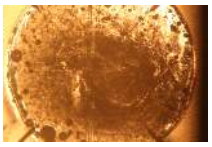
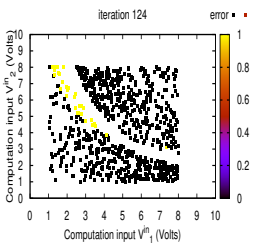


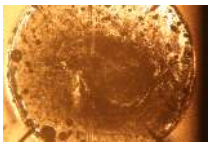
TABLE 5.4: Average verification $\overline{\Phi}_e^v(\%)$ errors for double training experiments on the same SWCNT/LC sample.

row	Once trained material			Twice trained material			
	SC $\mathbf{x}_1^*(SC)$	SC $\mathbf{V}=\mathbf{0}$	MC $\mathbf{x}_1^*(SC)$	MC $\mathbf{x}_2^*(MC)$	MC $\mathbf{V}^*=\mathbf{0}$	SC $\mathbf{R}_1^*,\mathbf{V}_1^*(SC)$	SC $\mathbf{V}=\mathbf{0}$
1	1.29	48.87	24.52	8.67	45.83	2.01	48.95
	MC $\mathbf{x}_1^*(MC)$	MC $\mathbf{V}=\mathbf{0}$	SC $\mathbf{x}_1^*(MC)$	SC $\mathbf{x}_2^*(SC)$	SC $\mathbf{V}^*=\mathbf{0}$	MC $\mathbf{R}_1^*,\mathbf{V}_1^*(MC)$	MC $\mathbf{V}=\mathbf{0}$
2	3.69	48.14	17.09	0.02	49.10	3.78	47.04
	V1C $\mathbf{x}_1^*(V1C)$	V1C $\mathbf{V}=\mathbf{0}$	SC $\mathbf{x}_1^*(V1C)$	SC $\mathbf{x}_2^*(SC)$	SC $\mathbf{V}^*=\mathbf{0}$	V1C $\mathbf{R}_1^*,\mathbf{V}_1^*(V1C)$	V1C $\mathbf{V}=\mathbf{0}$
3	0.29	0.29	51.16	1.32	49.48	5.03	6.73
	SC $\mathbf{x}_1^*(SC)$	SC $\mathbf{V}=\mathbf{0}$	V1C $\mathbf{x}_1^*(SC)$	V1C $\mathbf{x}_2^*(V1C)$	V1C $\mathbf{V}^*=\mathbf{0}$	SC $\mathbf{R}_1^*,\mathbf{V}_1^*(SC)$	SC $\mathbf{V}=\mathbf{0}$
4	0.69	49.43	31.02	0.37	0.41	21.19	49.45
	V1C $\mathbf{x}_1^*(V1C)$	V1C $\mathbf{V}=\mathbf{0}$	NLC $\mathbf{x}_1^*(V1C)$	NLC $\mathbf{x}_2^*(NLC)$	NLC $\mathbf{V}^*=\mathbf{0}$	V1C $\mathbf{R}_1^*,\mathbf{V}_1^*(V1C)$	V1C $\mathbf{V}=\mathbf{0}$
5	2.00	2.36	14.54	0.20	21.74	7.85	9.29
	NLC $\mathbf{x}_1^*(NLC)$	NLC $\mathbf{V}=\mathbf{0}$	V1C $\mathbf{x}_1^*(NLC)$	V1C $\mathbf{x}_2^*(V1C)$	V1C $\mathbf{V}^*=\mathbf{0}$	NLC $\mathbf{R}_1^*,\mathbf{V}_1^*(NLC)$	NLC $\mathbf{V}=\mathbf{0}$
6	1.19	2.35	10.76	0.00	0.76	3.39	5.78

produced when training for SC was not able, on average, to classify MC's verification instances with precision, resulting in a 24.52% error. The once-trained sample is subjected to a second DE training and $\mathbf{V}_\mathbf{t}^\mathcal{C}$(MC). The double-trained sample is now able to classify MC's verification instances with 8.67% error using the new solution $\mathbf{x}_\mathbf{2}^*$(MC). This is 5.35% higher than the optimum, but 15.85% lower than with $\mathbf{x}_\mathbf{1}^*$(SC). This suggests that the sample has been reconfigured by the algorithm for the new problem (MC). The ability of the material to retain the memory of the original problem (SC) was subsequently investigated. The sixth column of row 1, Table 5.4 shows that this is partly the case. The classification error for the SC problem when the full solution $\mathbf{x}^*$ was applied was 1.29%. When the partial solution $\mathbf{R}_1^*, \mathbf{V}_1^*$ is applied to the doubly-trained samples, it becomes 2.01%. Retraining has resulted in an increased of only 0.72% in SC verification error.

In the second set of experiments, for which results are presented in row 2 of Table 5.4, the reverse experiment is considered, i.e. MC-SC. In this case, SWCNT/LC samples are first trained to classify instances from the slightly more complex MC dataset, and the problem's optimum is reached. When tested against the previously unseen instances from the verification dataset, the error becomes 3.69% which is 0.36% above the problem's optimum. The error is 48.14% when no configuration voltages are applied, which is, once again, consistent with previous results. The solution produced by DE, $\mathbf{x}_\mathbf{1}^*$(MC) is not good enough to classify both MC and SC unseen instances, resulting in 17.09%



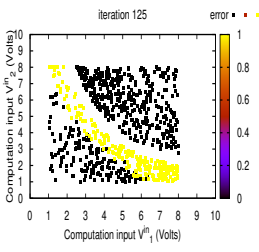


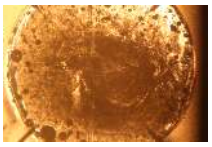
error. It must be noted that this is lower than when the less complex dataset was used first. Retraining of the sample by DE produces a new solution $\mathbf{x_2^*}$(SC) and verification error of 0.02%. The sample randomly classifies instances when it is not sent $\mathbf{V_2^*(SC)}$, but only two out of $K_v(SC) = 4000$ verification instances have been misclassified when the full solution is used. In addition, the double-trained sample is still able to solve the original MC problem, with a loss of 0.1% in accuracy when $\mathbf{V_1^*(MC)}$ and $\mathbf{R_1^*}$(MC) are used with $\mathbf{V_v^{\mathcal{C}}}$(MC).

A number of observations can be drawn from the results of the two sequences of experiments involving the SC and MC datasets. First, DE was able to reconfigure a SWCNT/LC sample which had been trained to classify instances from one dataset. Retraining produced a relatively accurate classifier. Secondly, the original solution was partly retained, despite modifications in the material produced by the DE-controlled electric field applied during training [8]. This translated in a low increase of the verification error post-retraining for the first dataset. Finally, the accuracy produced by the classifier was higher in the sequence of experiments where MC, the more complex dataset, was sent before SC. Here, a correlation appears between the complexity of the datasets used and the order in which the samples are trained.

Rows three and four of Table 5.4 present the results obtained using V1C and SC, in this order and the reverse. Once again these results are averaged over three experiments. There is more difference in the structure of the two datasets than between SC and MC. In addition, training with SC is performed using the scheme $\mathcal{S}_{\mathcal{C}}^{(1)}$, whilst $\mathcal{S}_{\mathcal{C}}^{(2)}$ is used with V1C. The un-configured SWCNT/LC sample trained with V1C data is able to classify 99.93% of the training instances. The solution $\mathbf{x_1^*(V1C)}$ generalises well, with 0.29% verification error. The contribution of the material state in this solution is important and the error remains 0.29% when no $\mathbf{V_1^*(V1C)}$ is applied. This first solution randomly classifies the SC verification instances. A second training is thus performed. The reconfigured sample produces a new solution $\mathbf{x_2^*(SC)}$ which reduces the verification error to 1.32% for SC. The original material state has been partly destroyed by the second training, with the V1C error with and without configuration voltages increasing by 4.74% and 6.44%, respectively.

When SC is used first, training results in a material able to classify the verification data with 0.69% error. This does not generalise well when no configuration voltages

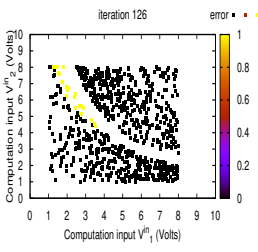


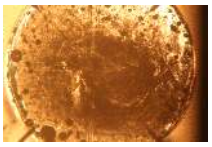


are use, or when they are used but with the V1C dataset (31.02%). The latter's training instances are used in a subsequent training. A near optimal 0.37% verification error is achieved using the second solution $\mathbf{x_2^*(V1C)}$. SWCNT structures have been modified by DE in such a way that the error increases of only 0.04% when no configuration voltages are applied. This however, resulted in a change of the material state such that the original solution was lost and sending $\mathbf{V_1^*(SC)}$ and $\mathbf{R_1^*(SC)}$ produced $21.19\%s$ verification error.

This set of experiments reinforces the observations regarding the SC/MC retraining. Despite a more important difference between the datasets, and a different implementation of $\mathcal{S}_\mathcal{C}$, samples can be reconfigured. The complexity of the dataset has an impact on the results over both training: when SC is sent first, both SC and V1C's verification results are better than when V1C is used for first. However, samples reconfigured to classify V1C instances were no longer able to solve the SC problem with high accuracy. This can be explained using the hypothesis proposed earlier and in [13]. The building of stable SWCNT structures by DE in its search for an optimal solution for V1C, has destroyed the original state which favoured SC. The search for a solution to SC does not modify the material state as much, hence a lower loss of accuracy for the post-retraining V1C verification results.

The last two rows, five and six, compare results obtained when using the most complex artificial dataset, NLC, with the simplest one, V1C. Starting with experiments where V1C is sent first, the DE algorithm is able, on average, to bring SWCNT/LC mixture into a state where it is able to correctly assign 100% of $V1C$ training data. Results presented in row 5 of Table 5.4 show that using the optimum set of decision variables 98% of verification can be correctly classified. When $\mathbf{V}^*$ is set to zero and the verification data are sent to the trained material, the error increases by only 0.36%. It can be deduced that the SWCNT structures evolved during training remained stable. The optimal solution for V1C, $\mathbf{x}_1^*(V1C)$, does not generalise to the NLC verification dataset. The error increases by 14.54% compared to an optimum NLC solution. DE is used to subject the SWCNT/LC to a new training, using the NLC dataset. Results are near optimal, with $\overline{\Phi_e^v} = 0.2\%$. The verification error is 21.74% when no configuration voltages are applied to the doubly trained material. Finally, the results when V1C is reapplied have increased to become 7.85% and 9.29%. The original solution is not fully recovered. The second





training affected the material state produced by the first.

Results for the reverse experiment, where the hyperbola-separated dataset was used first, followed by its diagonally separated counterpart are presented in the last row of Table 5.4. Verification tests with $\mathbf{x}_1^*$(NLC) produce an average of 1.19% error whilst verification increases to 2.35% when no configuration voltages are used. The first solution, $\mathbf{x}_1^*$(NLC), partly solves the V1C problem, but not well enough for further training to be redundant. Retraining the SWCNT/LC blend for the less complex V1C dataset results in 0% training (not shown in the table) and verification errors, with an increase of 0.76% when $\mathbf{V}^* = 0$. Re-using NLC instances and the original variable $\mathbf{R}_1^*$(NLC) with and without $\mathbf{V}_1^*$(NLC) results in 3.39% and 5.78% verification errors, respectively.

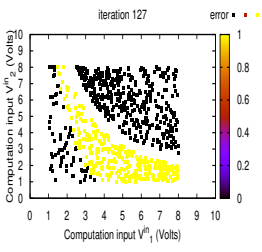
Here, training for the simple problem first helped in finding good solutions for the more complex problem. However, after finding a solution for the more complex problem, the state had changed in such a way that the original solution had been partly destroyed, producing verification errors higher by 5.85% and 6.93% with and without configuration voltages, respectively. It was also possible to achieve good solutions for the more complex problem starting from an un-configured material. The difference in this case is that the second training had less effect on the original solution, for which the verification errors increased by 2.2% and 3.43% with and without configuration voltages, respectively.

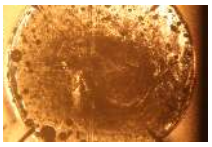
Results and discussion from all retraining experiments suggest that:

1) It is possible to reconfigure SWCNT/LC samples from a state where it is able to classify instances from one dataset into a state where it can do so for another. The verification error for the reconfigured samples was similar to the average first training error. In all cases, the untrained material produced $\overline{\Phi}_e^v \simeq 50\%$, whilst the doubly trained material's solution achieved, at a maximum, $\overline{\Phi}_e^v = 21.19\%$.

2) The complexity of a dataset affects the DE's search and the resulting transformation of samples. Training a sample first with the more complex dataset produced a material state that favoured the subsequent search for a solution to the simpler problem. In addition, if the more complex dataset was used first, the doubly-trained device generalised better for both datasets' instances.

3) The *in materio* search performed by the algorithm has an impact on the ability of





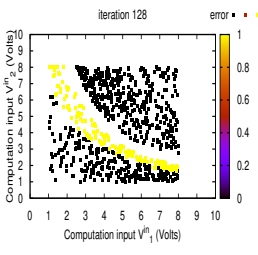
reconfigured samples to classify more than one dataset. In MC-SC experiments, subjecting samples to a second training resulted in a minimal loss of classification accuracy. This loss was more important when using V1C on SC-trained samples. The modifications induced in the material during DE's search partly destroyed the original solution, despite V1C being slightly simpler. On the other hand, in the V1C-NLC experiments the loss of accuracy post V1C training was less important. It appears that the material state produced during NLC search was more stable than for SC.

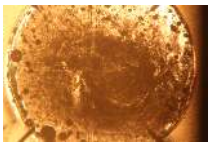
5.6 Stability of Solutions

Using SWCNT/LC samples, tests have been performed to define the length of time the material can be left - post training - before its problem-solving state is destroyed. Verification with configuration voltages was performed ten seconds after training, followed by tests with no configuration voltages, one minute after training ended. The difference in results between these two runs was 1.7 %. After eight hours, verification was repeated with no configuration voltages, resulting in an average increase of 6% error from the original solution. Approximately 4.3% of the solution's accuracy has been lost over 8hrs. Further testing over set periods of time would need to be undertaken in order to provide a good estimate of solution deterioration over time. However, the aim here was to determine a general maximum time before the solution would be completely deteriorated according to the time taken for a SWCNT/epoxy composite to solidify under UV light. These experiments are reported in the next chapter and are the logical step following the discussions reported here.

5.7 Summary of Results and Conclusions

Experimental results reported support the conclusion that EiM based on the SWCNT/LC composite is able to yield a system capable of performing reproducible binary classification. It is also observed that the degree of the material contribution to the overall solution, as complemented by the decision variables (configuration voltages and auxiliary quantities), depends on the classification problem's complexity. When this complexity is low, the optimisation algorithm tends to converge to solutions where the optimum configuration voltages have a dominant role in the classification decision. On the other hand,





materials trained on the more complex problems see little increase in error when the configuration voltages are not applied. This implies that for more complex problems, the search for a solution further explores the subspace of nanotube network formations. The emerging macro behaviour of the material's conductivity as affected by those networks can eventually be used for performing the classification task.

Another set of experiments, where a successful training was repeated exactly on different initially unconfigured samples shows that the process can be automated to a high degree. Successful sequences of applied configuration voltages tend to be more effective when the solutions obtained are dominated by the material rather than the decision variables. Their transferability is therefore better for more complex problems.

The last results presented indicate that it is possible to enable one material sample to solve two different classification problems, using different sets of decision variables. The effectiveness of the double training depends on the ordering of the problems for which the sample is trained. Training the material following a descending problem complexity sequence results in better classifier performance. Converging to good minimum solutions first allows the inclusion within the material structure of information required for addressing a less complex problem.

In summary, the contribution of this chapter is three-fold: 1) providing a new measure of confidence for the evolved classifiers, based on electrical characteristics (current outputs); 2) demonstrating the impact of EiM on training of liquid SWCNT/LC composites in terms of device reproducibility, reconfigurability and memory; and 3) a first attempt at partially automating the evolution process.

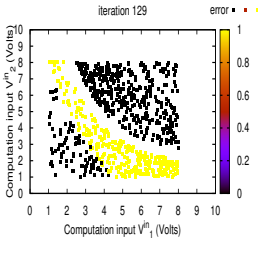
Bibliography

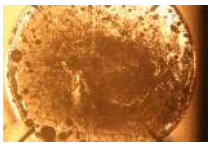
[1] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

[2] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Information processing & management*, vol. 45, no. 4, pp. 427–437, 2009.

[3] C. Ferri, J. Hernández-Orallo, and R. Modroiu, "An experimental comparison of performance measures for classification," *Pattern Recognition Letters*, vol. 30, no. 1, pp. 27–38, 2009.

[4] M. Dale, S. Stepney, J. F. Miller, and M. Trefzer, "Reservoir computing in materio: A computational framework for in materio computing," in *2017 International Joint Conference on Neural Networks (IJCNN)*, pp. 2178–2185, IEEE, 2017.





[5] C. Parker, "An analysis of performance measures for binary classifiers," in *2011 IEEE 11th International Conference on Data Mining*, pp. 517–526, IEEE, 2011.

[6] A. Mandelbaum and D. Weinshall, "Distance-based confidence score for neural network classifiers," *arXiv preprint arXiv:1709.09844*, 2017.

[7] M. Poggi, F. Tosi, and S. Mattoccia, "Quantitative evaluation of confidence measures in a machine learning world," in *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 5238–5247, 2017.

[8] D. Volpati, M. K. Massey, D. Johnson, A. Kotsialos, F. Qaiser, C. Pearson, K. Coleman, G. Tiburzi, D. A. Zeze, and M. C. Petty, "Exploring the alignment of carbon nanotubes dispersed in a liquid crystal matrix using coplanar electrodes," *Journal of Applied Physics*, vol. 117, no. 12, p. 125303, 2015.

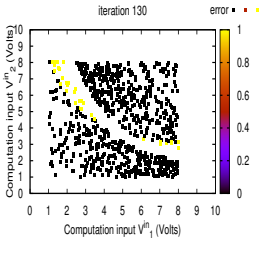
[9] M. E. H. Pedersen, "Good parameters for differential evolution," tech. rep., Technical report, Hvass Computer Science Laboratories, 2010.

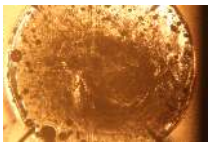
[10] E. Vissol-Gaudin, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, C. Groves, and M. C. Petty, "Data classification using carbon-nanotubes and evolutionary algorithms," *International Conference on Parallel Problem Solving from Nature*, pp. 644–654, 2016.

[11] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[12] J. F. Miller, S. L. Harding, and G. Tufte, "Evolution-in-materio: evolving computation in materials," *Evolutionary Intelligence*, vol. 7, no. 1, pp. 49–67, 2014.

[13] E. Vissol-Gaudin, A. Kotsialos, C. Groves, C. Pearson, D. A. Zeze, and M. C. Petty, "Computing based on material training: Application to binary classification problems," *2017 IEEE International Conference on Rebooting Computing (ICRC)*, pp. 1–8, 2017.





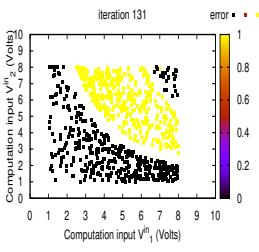
Chapter 6

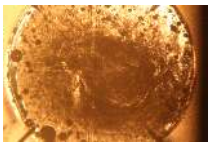
Classification in SWCNT / Epoxy Composites and Device Encapsulation

6.1 General Overview

In the previous chapters, it has been observed that evolutionary algorithms (EAs) were capable of evolving samples of a single-walled-carbon nanotube (SWCNT) / liquid crystal (LC) composite, resulting in a material state where the classification of data was possible. For the specific datasets used, results were better than those obtained with solid SWCNT/ poly(butyl metaacrylate) (PBMA) composites using exactly the same implementation, and comparable to those presented in [1]. In the later case, the samples were nominally the same, but the hardware and algorithms used were different.

Interestingly, it was observed in Chapter 5 that the SWCNT/LC samples' evolved state allowed data to be classified without the need for applied configuration voltages. In other words, the training had evolved a classifying SWCNT/LC device rather than just optimised a set of voltages making a SWCNT/LC sample behave as a classifier. In addition, the classifying state of the SWCNT/LC evolved samples was stable for a number of hours with little deterioration in the classification error. This is different from the stability of solution observed in other experiments were a dynamic material was used. In the latter, a high fitness, i.e. low error, obtained by a robot controller evolved in a liquid crystal display (LCD) [2] could only be reproduced for about thirty seconds.

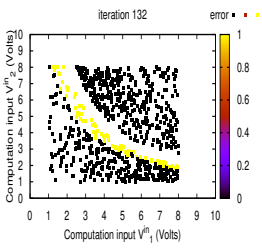


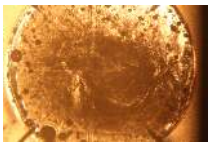


Results obtained with the SWCNT/LC composite suggested that the ability to classify data without configuration voltages resulted from the SWCNT structures produced by the algorithm's search process. It was observed that this ability tended to depend on the complexity of the classification problem which the samples were evolved to solve. The structures are reminiscent of the iron thread grown by Gordon Pask in ferrous sulfate ($FeSO_4$) to perform tone discrimination [3], albeit in the case of the SWCNT/LC composite, the structures are the results of nanotube *bundling* and *rearrangement*, rather than the *growth* of intertwined metallic wires.

It must be noted that despite an initial solution stability longer than that observed in any other dynamic material used in evolution in materio (EiM), some deterioration in the classification error obtained by evolving SWCNT/LC samples was observed after a number of hours. More importantly, because of the dynamic nature of the LC matrix, large external tampering, such as shaking or dropping of the sample, resulted in a complete modification of the material's bulk morphology, or even destruction of the sample. Based on these considerations, it was decided to produce hybrid liquid/solid SWCNT-based composites, with the aim of keeping the benefits of the search space offered by the liquid state whilst mitigating the impact on long term stability, as well as handling and storage requirements. Such material would be trained when in a liquid state, allowing the modification of SWCNT-based structures by the EA until a computation inducing state is reached. The evolved device would then be cured. The curing can be considered as a process of electronic device encapsulation. The length of curing necessary for these hybrid composites to reach a solid state would need to be smaller than the time limit after which the computation error begins to deteriorate.

This chapter presents the first, and preliminary, investigations relating to the evolution of SWCNT/epoxy (LP655 and NO81) composites into computational devices. Instead of LCs, a two-part UV-cure epoxy resin was used as a matrix for the SWCNT dispersion. The general ability of this material to be evolved using EiM to solve computational problems is first assessed, and results are compared with those obtained with the SWCNT/LC and SWCNT/PBMA composites. The possibility for stable SWCNT structures to be evolved is then discussed along with the deterioration of the solution due to time and to the curing process. A summary of the results and possible avenues for further work concludes this chapter.





6.2 Classifying Data with SWCNT/Epoxy Samples in Liquid State

Two different epoxies, LP655 [4] and NO81 [5] were mixed with SWCNTs. The characteristics of these composites have been discussed in Chapter 2. Their electrical behaviour followed the non-linear I/V curve characteristic of devices chosen for EiM and their current levels were significantly higher than the hardware noise. The viscosity of both epoxies in liquid state was lower than that of the solidified PBMA, thereby enabling the SWCNTs to move within the composite under an applied electric field. In addition, the SWCNT/epoxy composites were able to solidify completely with the added SWCNT load, up to a given SWCNT concentration.

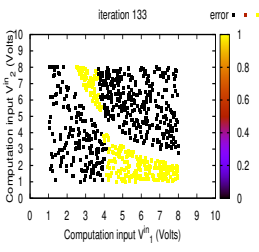
The first experiments reported were concerned with testing whether the SWCNT/ epoxy samples in liquid state are capable of being evolved to solve computational problems. Synthetic binary classification problems (BCPs) were chosen as a mean of comparison with the nanotube-based materials studied previously.

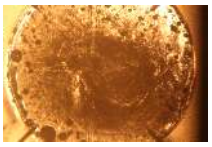
6.2.1 SWCNT/LP655 Composite

The SWCNT/LP655 composite was used in the first set of experiments. As proof-of-concept for this new material, the DE algorithm, implemented with the parameters presented in Chapter 3, was used to find solutions to the simplest binary classification problem (BCP). In other words DE's task was to induce a material state favouring the correct classification of all instances from the SC dataset described in Chapter 4.

Comparing SWCNT/LP655 performance with other nanotube-based samples

Results reported in Table 6.1 are averaged over 3 experiments for the SWCNT/ LP655 and SWCNT/PBMA samples and averaged over 20 experiments for the SWCNT/LC composite. In these experiments, the SWCNT/LP655 were evolved in their un-cured state. The optimum training (Φ_e^{t*}) and average verification errors ($\overline{\Phi}_e^{v}$) assess the accuracy of evolved samples. The average convergence rate towards a solution is represented by the number of iterations, λ^*, necessary to achieve Φ_e^{t*} and averaged over experiments. A measure of the over-fitting of the SWCNT/LC classifier to training data is given by the absolute difference between the optimum training and verification errors averaged over



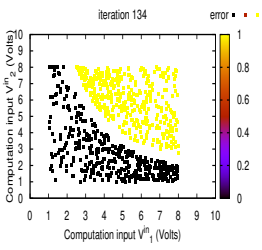


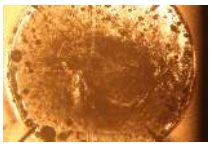
experiments ($\overline{d_\Phi} = |\Phi_e^{t*}\text{-}\Phi_e^{v*}|$). In the liquid samples, this difference can also be a measure of the changes in material morphology since the optimum solution was obtained, i.e. a measure of the stability of the SWCNT structures evolved during DE's search. Since the number of experiments undertaken with SWCNT/PBMA and SWCNT/LP655 was small, differences between the metrics presented in Table 6.1 could be attributed to under-sampling. In order to provide an additional support for the discussion based on these differences, the significance of these differences was assessed using the two sided Mann-Whitney U-test. The qualitative result of this test, i.e. significant/not significant is reported in Table 6.1 for the SWCNT/LP655 compared to the SWCNT/PBMA on one hand, and SWCNT/LP655 compared to SWCNT/LC on the other. The significance of the difference between the average results of the SWCNT/LC and SWCNT/PBMA are not reported.

It can be seen in Table 6.1 that the optimum solution to the SC problem, resulting in 0% error, was not found by the DE algorithm when training SWCNT/LP655 samples. On average, the optimum training error was $\Phi_e^{t,*} = 2.87\%$. This is far from the 50% error that would be obtained if the SC dataset's instances were classified randomly, and from the training error generally obtained with the control materials. In addition, the training error was not significantly higher than that obtained with SWCNT/LC samples (0.26%), whilst significantly lower than the SWCNT/PBMA training error (18.83%). The number of iterations, λ^*, needed for the best training error to be obtained by SWCNT/LP655 composite was, on average, nearly the same as SWCNT/LC samples.

TABLE 6.1: Comparing algorithm performance in solving the SC problem with SWCNT/PBMA, SWCNT/LP655 and SWCNT/LC.

SC dataset	SWCNT/PBMA	SWCNT/LP655	SWCNT/LC
Φ_e^{t*}	18.83%	2.87%	0.26%
U-test	significant	not significant	
λ^*	1021.67	93.67	92.75
U-test	not significant	not significant	
$\overline{\Phi}_e^v$	36.41%	34.72%	1.30%
U-test	not significant	significant	
d_Φ	17.17%	32.13%	0.863%
U-test	not significant	significant	





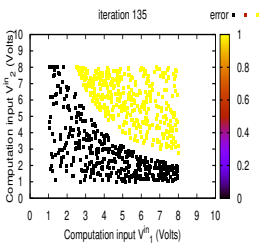
These first results would suggest un-cured SWCNT/LP655 samples to be a good candidate for EiM. However, both in terms of average verification error, and of the difference between training and verification error which illustrates how well the evolved composites generalise, results for the SWCNT/LP655 composite were significantly higher than those obtained with SWCNT/LC. When comparing SWCNT/LP655 with SWCNT/PBMA, neither average verification error, nor the difference between training and verification are significantly different. However, for both materials, these measures suggest poor performance of the evolved classifiers.

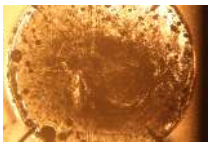
It is difficult to determine why results obtained with SWCNT/LP655 samples were not as good as those obtained with the other two composites. The issue could arise from the algorithm's inability to find solutions to the problem at hand in the five experiments undertaken. However, the analysis of the reproducibility of results using the DE algorithm suggests otherwise. Another possible reason for the poor results obtained with the SWCNT/LP655 composite would be that the search performed by the algorithm created important modifications in the samples' structure between the iteration λ^*, where the optimum Φ_e^{t*} was achieved and the verification tests when $\overline{\Phi_e^v}$ was produced.

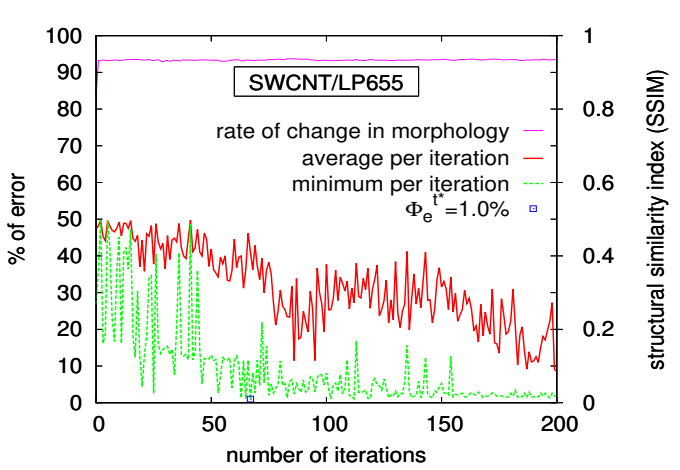
Rate of change in material morphology and training performance

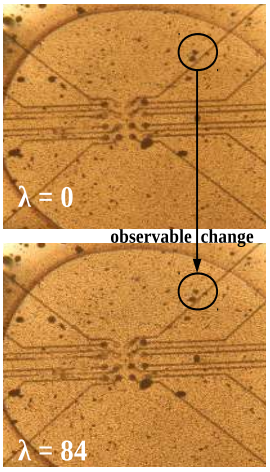
In order to investigate whether solution deterioration was caused by important changes in the material under DE's configuration voltages, the rate of change in morphology between iterations was computed. It is illustrated in the left graphs of Figure 6.1(a) and (b) by the variations in the structural similarity index (SSIM) [6]. The SSIM is used to compare each photograph with the next (one photograph was taken per iteration). The SSIM value decreases as the level of change increases. The convergence of the objective function, in terms of average and minimum training error per iteration, as well as the iteration at which the optimum was achieved is also presented in this figure.

It can be observed from both graphs that the rate of change (RoC) in the morphology of the SWCNT/LP655 sample during the training presented in Figure 6.1(a) is constant across iterations, at a level of $\approx$ 0.93 SSIM. A similar behaviour is observed in the SWCNT/LC sample in Figure 6.1(b), however, in the latter, the SSIM is constantly $\approx$ 0.83. Figure 6.1(c) illustrates the configuration voltages trajectories for both composites over 100 iterations, since $\lambda = 84$ is the point at which DE search terminated for the

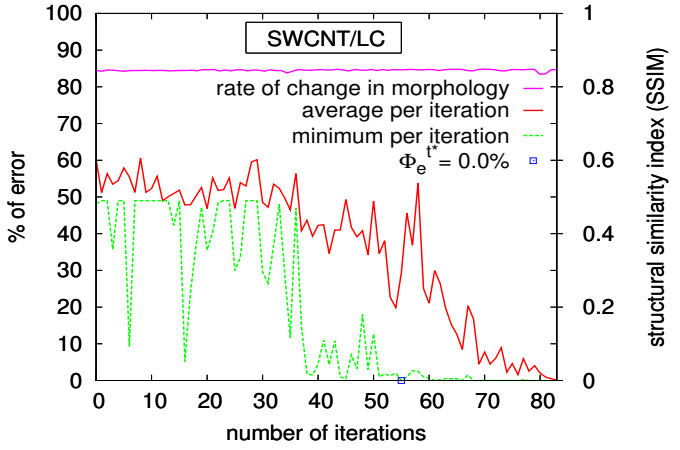


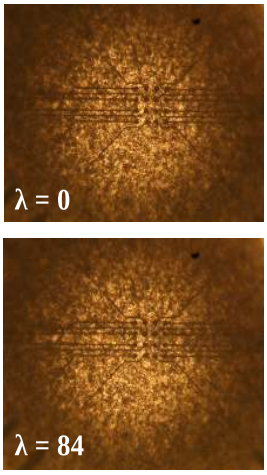




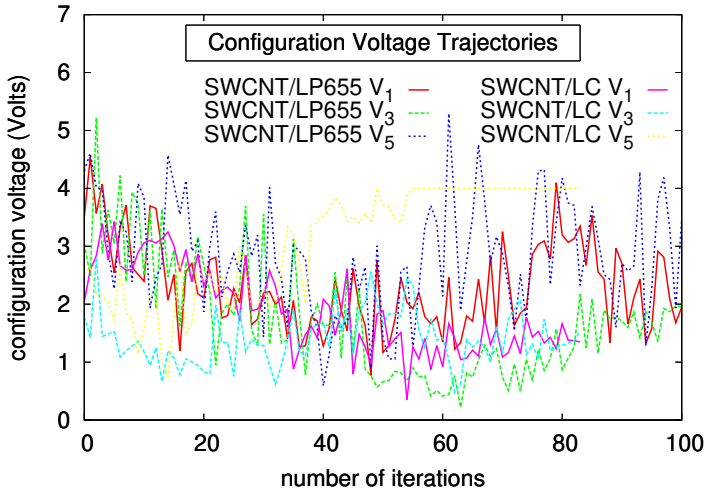


(a)



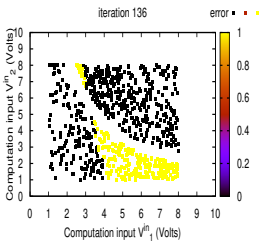


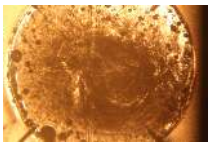
(b)



(c)

FIGURE 6.1: Average and minimum classification error per iteration, along with the rate of change observed from the surface of the material at micro-scale during training for the SC problem, with (a) a SWCNT/LP655 sample and (b) a SWCNT/LC sample. The configuration voltage trajectories for the two samples are illustrated in (c)





SWCNT/LC sample. It can be observed that the trajectories of the voltages in each sample are effectively indistinguishable. This suggests that the difference in SSIM across iterations between the two samples was due to the higher viscosity of the SWCNT/LP655 rather than a difference in the DE search.

Whilst modifications to the morphology were minimal in both samples (see right hand side of fig. 6.1(a) and (b)), the SWCNT/LC sample's morphology changes more during training than that of SWCNT/LP655. The deterioration in classification error between training and verification, $\overline{d_\Phi}$, reported in Table 6.1, can therefore be attributed to the inability of the algorithm to find suitable solutions to the SC problem, rather than changes that would be induced in the sample between $\lambda = \lambda^*$ and $\lambda = \Lambda$.

Results reported in Table 6.1 and the behaviour observed in Figure 6.1 suggest that the SWCNT/LP655 did not constitute a suitable material for EiM as implemented here.

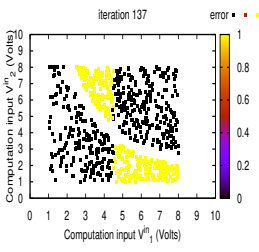
6.2.2 SWCNT/NO81 Composite

Results obtained with the SWCNT/LP655 were far from optimum, an issue which has been attributed to the high viscosity of the samples. A different epoxy, NO81 [5], was therefore used in the second set of SWCNT/ epoxy experiments. It presents similar electrical characteristics as the LP655 epoxy, with a lower viscosity. Instead of SC, the two problems to solve with the SWCNT/NO81 composite were V1C and NLC (see Chapter 4). The choice of these problems was motivated by the discussion on solution stability and retraining, reported in Chapter 5.

Comparing SWCNT/NO81 performance with other nanotube-based samples

The metrics used in Table 6.2 for assessing and comparing the quality of DE training and evolved solution are the same as those used in Table 6.1. Results obtained with SWCNT/NO81 are compared with those obtained with the SWCNT/LC for the V1C problem and both SWCNT/LC and SWCNT/PBMA for the NLC problem. In all cases, the results presented for the SWCNT/NO81 were obtained whilst pre-curing of the material.

It can be observed from the first row of Table 6.2 that DE training was capable of bringing the SWCNT/NO81 composite into a state where an error close to the V1C problem's minimum could be found, with consistency across experiments. In one case,



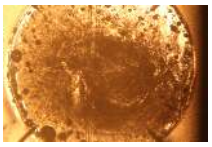


TABLE 6.2: Comparing algorithm performance in solving the V1C and NLC problems with SWCNT/NO81, SWCNT/LC and SWCNT/PBMA.

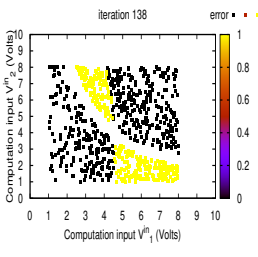
	V1C dataset		NLC dataset		
	SWCNT/NO81	SWCNT/LC	SWCNT/PBMA	SWCNT/NO81	SWCNT/LC
$\Phi_e^{t,*}$	2.03%	0.21%	5.23%	8.74%	0.78%
U-test	not significant		not significant	not significant	
λ^*	83.67	82.4	75	122	212
U-test	not significant		not significant	not significant	
$\overline{\Phi_e^v}$	7.94%	1.31%	19.88%	11.59%	5.79%
U-test	not significant		not significant	not significant	
$\overline{d_\Phi}$	6.76%	0.74%	14.13%	1.96%	3.34%
U-test	not significant		not significant	not significant	

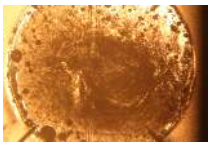
a 0% training error was achieved, and on average over all experiments, $\Phi_e^{t,*} = 2.03\%$. This is higher than the average training error obtained across experiments in SWCNT/LC ($\Phi_e^{t,*} = 0.21\%$), but not by a significant amount. For both composites, the training optima were, on average, found after a very similar number of iterations: $\lambda^* \simeq 83$. At this point, results suggest that the SWCNT/NO81 composite would be a good candidate for EiM.

The verification results confirm this hypothesis to an extent. Across experiments using different samples of the SWCNT/NO81 composite, $\Phi_e^{t,*} = 7.94\%$. Unlike solutions found by the DE algorithm with SWCNT/LP655, solutions found with SWCNT/NO81 were able to classify with relative accuracy the unseen instances from the verification set of V1C. In addition, whilst the verification error was higher than that obtained with the SWCNT/LC samples, it was not significantly so. The measure of over-fit, $\overline{d_\Phi}$, suggests that the quality of the solutions obtained in SWCNT/NO81 was not as good as that obtained with SWCNT/LC samples. However, the SWCNT/NO81 did demonstrate the potential of being able to solve the V1C problem through algorithm training.

The second set of experiments was undertaken with the more complex NLC dataset. As expected, compared to the less complex V1C problem, solutions found resulted in a higher percentage of error for all materials. The training results presented in Table 6.2 shows that DE did not, on average, find the optimum solution for the NLC problem. However, for each material, $\Phi_e^{t,*} = 0\%$ was achieved in at least one experiment.

The training error for the NLC problem obtained on average across un-cured SWCNT/NO81 samples, $\Phi_e^{t,*} = 8.74\%$, was higher than the $\Phi_e^{t,*} = 0.78\%$ and $\Phi_e^{t,*} = 5.23\%$



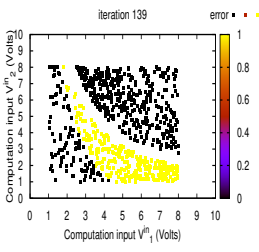


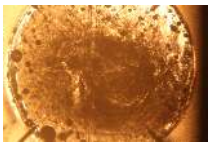
obtained with SWCNT/LC and SWCNT/PBMA samples, respectively. However, the differences were not statistically significant and neither were the differences in λ^*. On average the optimum solution to the NLC problem, resulting in $\Phi_e^{t,*}$, was found faster with SWCNT/NO81 than with the SWCNT/LC, and slower than with SWCNT/PBMA. These differences can be attributed to the important variations in λ^* observed across experiments.

The verification error for the SWCNT/NO81 samples trained to solve the NLC problem was 11.59%. This is significantly lower than the 50% error obtained in control experiments and in SWCNT/NO81 samples prior to training. The verification error in evolved SWCNT/NO81 samples was also significantly lower than that obtained with the SWCNT/LP655 for the less complex SC problem. This confirms that the NO81 epoxy was a better choice under the experimental implementation used here. Compared to the other two materials trained to solve the NLC problem, the SWCNT/NO81 verification error was almost half that obtained with the solid SWCNT/PBMA composite but it was almost twice as high as SWCNT/LC's verification error. Neither difference was statistically significant however, and a wide spread across the verification errors obtained with SWCNT/NO81 was observed, suggesting a problem with reproducibility in the experimental implementation used here.

In the SWCNT/PBMA samples, the difference between training and verification errors, $\overline{d_\Phi} = 14.13\%$ was mainly the result of outliers. The presence of outliers can be attributed to the algorithm's inability to consistently find solutions that generalise well. Potential explanations for the 3.34% obtained with SWCNT/LC have been discussed in details in previous chapters. In this case, results suggested that if the optimum training error was achieved in the last iteration, the difference between training and verification errors could be attributed to the same cause as for SWCNT/PBMA. If $\lambda^* \neq \Lambda$, it would be combined with the fact that the evolution of the material did not result in SWCNT structures stable enough to be unaffected by subsequent training.

Compared to both SWCNT/LC and SWCNT/PBMA, solutions obtained through evolution of the SWCNT/NO81 samples presented the lowest difference between training and verification errors, and low variance across experiments. In other words, the absolute difference between optimum training and verification errors, both averaged across experiments, was lowest in SWCNT/NO81. Since the liquid state of the SWCNT/NO81





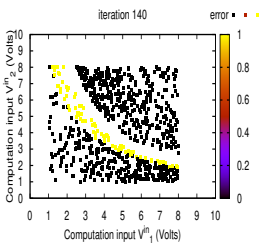
composite was closer to that of SWCNT/LC, it would be easy to suggest that the deterioration in error was due to the large changes in the material morphology across iterations. The results of these changes would be the destruction of the physical part of the solution evolved by DE during training. However, without more information, it is not possible to determine the validity of this hypothesis.

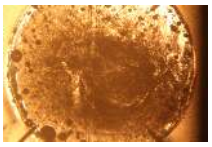
Rate of change in material morphology and training performance

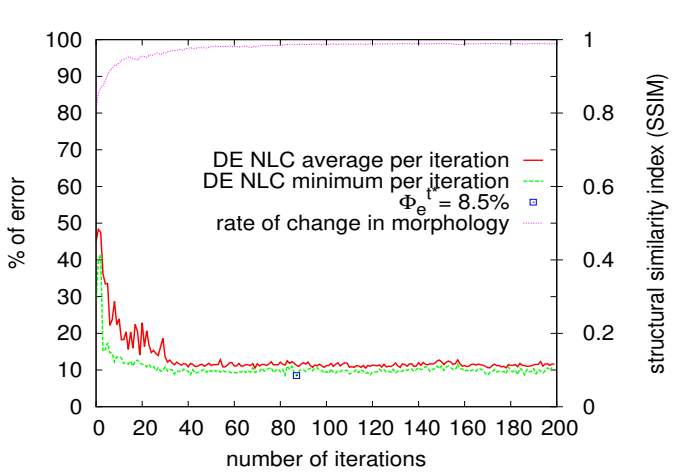
As for the SWCNT/LP655 composite, the rate of change in the morphology observed in SWCNT/NO81 samples between iterations was computed in order to investigate whether solution deterioration was caused by important changes in the material under DE's configuration voltages. It is illustrated in the left graph of Figure 6.2 (a) by the variations in the SSIM [6], along with the objective function, in terms of average and minimum training error per iteration. Photographs of the material at different iterations are included in the right hand side of fig. 6.2 (a). The same information is reported in Figure 6.2 (b) for the SWCNT/LC composite. Figure 6.2 (c) presents three of the configuration voltage trajectories that, for each material, produced the states resulting in the objective functions of fig. 6.2 (a) and (b) and the associated changes in morphology observed in the photographs.

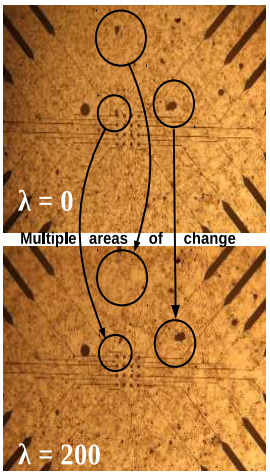
It can be observed from Figure 6.2(a) that the highest changes in morphology were produced in SWCNT/NO81 sample in the first few iterations. An SSIM of 0.82 was obtained when comparing the photographs taken during the first and second iterations. This is comparable to the SSIM achieved in SWCNT/LC during training, and lower that the SSIM observed during SWCNT/LP655 training. At the microscopic scale, changes in the material between the first and last iterations are more important in the SWCNT/NO81 sample presented in the left hand side of Figure 6.2(a) than in the SWCNT/LP655 sample from Figure 6.1(a), confirming that the NO81 epoxy allows SWCNTs to move under an applied electric field, more than the LP655.

The iterations during which the highest changes are observed correspond to those where the percentage in classification error was highest ($\simeq 50\%$, corresponding to random classifying of data). Following the first iterations, the rate of change decreases non linearly. This is illustrated in Figure 6.2(a) by a non-linear increase in SSIM to $\simeq 0.98$ after $\lambda = 40$. The convergence of the average and minimum error functions per iterations

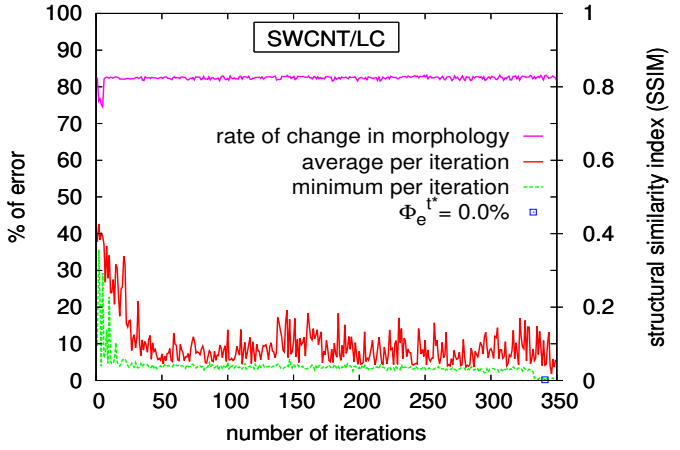


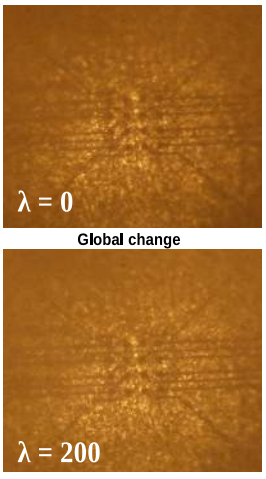




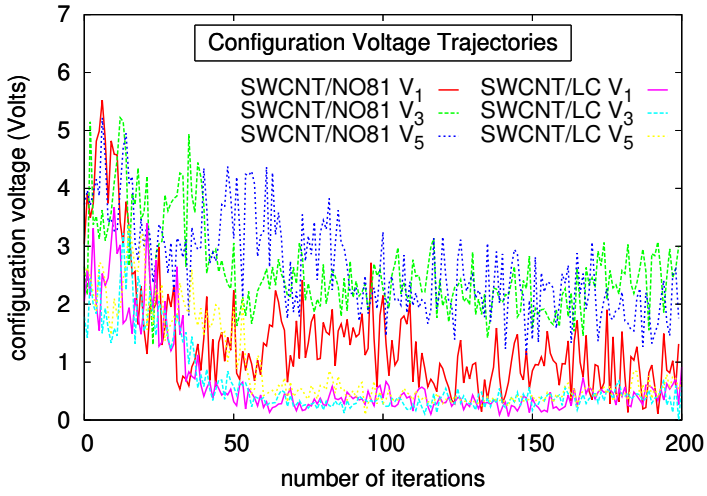


(a)



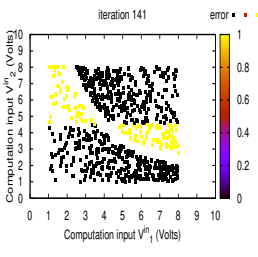


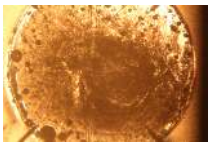
(b)



(c)

FIGURE 6.2: Average and minimum classification error per iteration, along with the rate of change observed from the surface of the material at micro-scale during training for the NLC problem, with (a) a SWCNT/NO81 sample and (b) a SWCNT/LC sample. The configuration voltage trajectories for the two samples are illustrated in (c)



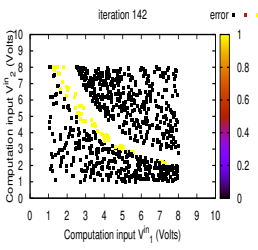


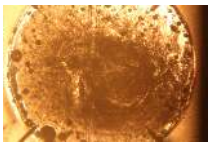
for the SWCNT/NO81 composite is nearly the inverse of the rate of change in material morphology. Both error functions decreased non-linearly towards 8.5% error and remained within 10% of the NLC problem's minimum past $\lambda = 40$. No better solution, resulting in a lower error, was found before the end of training. However, the solution obtained at iteration 83 is the one used in verification, since it produced the lowest error percentage achieved closer to the end of training. Using the last solution achieving the experiment's optimum supposedly ensures the least possible amount of variation in material state between that tested during verification, and the one which produced the lowest error.

The high amplitude of the configuration voltage trajectories between iterations $\lambda = 0$ to $\lambda = 40$, presented in Figure 6.2(c), suggests that a DE search focused exploration towards the start of the training process. The subsequent convergence of the configuration voltage trajectories towards smaller values suggest that once a solution resulting in an error within 10% of the problem's minimum was found, the algorithm exploited the search space around this solution. However, it appeared to be a local optima, and no better solution was found before the end of training.

The second graph on the right of Figure 6.2(b) illustrates the behaviour of the objective function and the rate of change in material morphology for a SWCNT/LC sample. It can be observed that the convergence of the objective function followed a course similar to that obtained in SWCNT/NO81 samples. The SSIM also behaved in a similar way: it was at its lowest during the first iterations, indicating large changes in the material at microscale when the error was highest and it increased as the algorithms converged towards better solutions to the NLC problem.

The advantage of the lower voltage levels which can result from a search focused on exploitation is that they produce the low variations in material morphology illustrated by a SSIM $\simeq 1$. Considering that a similar behaviour was observed in all experiments where SWCNT/NO81 samples were trained to solve the NLC problem, the minimal loss of accuracy between optimum training can be attributed to the negligible rate of change in morphology resulting from the algorithm's exploitation of the search space. This behaviour is also unfortunately the cause of important variations in error across experiments, since an algorithm trapped in the vicinity of a local optimum will likely be incapable of finding the problem's optimal solution.





In the case of the SWCNT/LC, Figure 6.2(b) shows that the SSIM remained around 0.84 as the algorithm settled into an exploitation phase resulting in lower levels of configuration voltage trajectories. This can be attributed to the fact that the SWCNT/LC composite has a lower viscosity than the SWCNT/NO81 composite. It is also a possible reason for the higher differences between training and verification results reported in Table 6.2. In this case, a low difference will only be achieved if the algorithm has evolved structures strong enough to resist the changes occurring between λ^* and Λ.

Results obtained with the SWCNT/NO81 composite, albeit requiring improvement, were not significantly different from the results obtained with the SWCNT/LC composite. This motivated further study into this material's stability, and its ability to retain solutions after being subjected to the curing process.

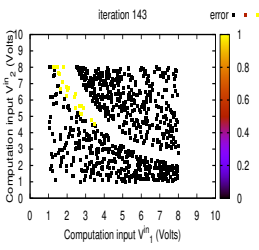
6.3 Stability of Solutions

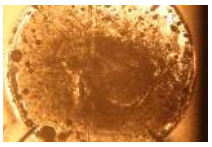
Results obtained with the SWCNT/LP655 composite are not analysed here since the level of error in this material was too high for it to be of interest in future experiments.

The previous section demonstrated the possibility to evolve SWCNT/NO81 samples into devices capable of solving two different classification problems. In addition, it was observed that the material morphology could be modified by an algorithm-controlled electric field to a similar extent as SWCNT/LC samples. Despite the fact that the SWCNT/NO81's morphology could be affected by DE's search, unlike SWCNT/PBMA composites, it was observed that a minimal deterioration in error between training and verification was generally observed across experiments.

Since the SWCNT/NO81 presents a low viscosity, the stability of the evolved solution, i.e. its ability to solve a computational problems after a given amount of time with minimal loss of accuracy, is a concern. A good stability is especially important since the aim of using this material is to cure evolved solutions. A poor stability would results in a loss in accuracy before the curing process might have began, and therefore the advantage (see Chapter 5) of the liquid SWCNT/NO81 composite over solid SWCNT/PBMA samples would be lost.

The stability of solutions evolved in SWCNT/NO81 has therefore been assessed, through two different means. The first consists in calculating the dispersion in error between the ten verification tests undertaken at the end of training (the average over these





tests was used in the previous sections). This value of dispersion, i.e. the standard deviation in results across Q tests, averaged over experiments, is reported as $\overline{\sigma(\phi_e^v)}$ in Table 6.3. The second consists in testing the solutions after four hours which is reported as $\Phi_e^v(\mathbf{x}^* + 4h)$ in the table. Four hours is more than the minimum time required in order to cure the SWCNT/NO81 samples at the nanotube concentration used. In addition, it was the critical time after which solutions obtained with SWCNT/LC had either deteriorated, or after which the change in error remained with 5%.

Results regarding training and verification errors previously reported are reproduced in Table 6.3, along with those obtained for three different metrics of solutions stability in evolved SWCNT/NO81 composite trained to solve the V1C and NLC problem.

TABLE 6.3: Stability of solution in evolved SWCNT/NO81 samples.

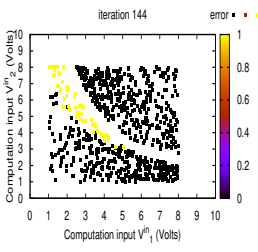
Datasets	$\Phi_e^{t,*}(\%)$	$\overline{\Phi}_e^v(\%)$	$\overline{\sigma(\phi_e^v)}(\%)$	$\Phi_e^v(\mathbf{x}^* + 4h)(\%)$
V1C	2.03	7.94	0.8232	9.823
NLC	8.74	11.59	0.4096	13.46

A low spread of error across the ten verification tests performed 5 minutes after the end of the training process can be observed for both problems in the table. This is illustrated by the standard deviation $\overline{\sigma(\Phi_e^v)}$, which is lowest for the NLC dataset, suggesting a high solution stability, although the verification error, $\overline{\Phi}_e^v$. was not optimal. With respect to the verification error a deterioration was observed when testing the evolved classifier four hours after the end of training, $\overline{\Phi}_e^v = 11.59\%$ vs $\Phi_e^v(\mathbf{x}^* + 4h) = 13.46\%$, although not to the extent of a complete return to the untrained material state where instances from the two datasets were classified at random, i.e. 50% error (see Chapter4, section4.2).

6.4 Curing Evolved SWCNT/NO81 Classifiers

The percentage of classification error obtained with SWCNT/LP655 samples was too high to be of interest and justify curing of the evolved material. On the contrary, the set of experiments for which results are reported in Table 6.3, suggests that attempting to cure evolved SWCNT/NO81 classifiers is not a redundant task.

The potential solution deterioration caused by the curing process, where the liquid composite becomes an insoluble, infusible polymer, is assessed here. Table 6.4 reports the verification error obtained when evolved liquid SWCNT/NO81 samples have been



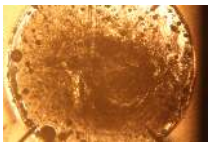


TABLE 6.4: Comparison of classifier performance before and after curing of SWCNT/NO81 composites evolved to solve the V1C and NLC problems.

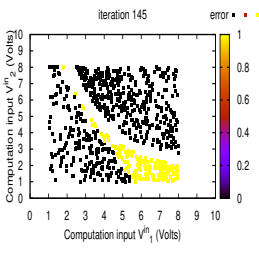
Datasets	V1C		NLC	
State	Liquid	Solid	Liquid	Solid
$\Phi_e^{t,*}$ (%)	2.03	N/A	8.74	N/A
$\overline{\Phi}_e^v$ (%)	7.94	11.52	11.59	27.77
$\Phi_e^v(\mathbf{x}^* + 4h)$ (%)	9.82	N/A	13.46	N/A
$\overline{\sigma(\Phi_e^v)}$ (%)	0.823	0.444	0.409	0.922

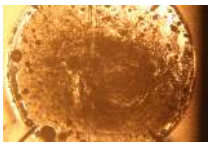
cured under UV light, compared with the error obtained in the same samples, pre-curing, i.e. when the SWCNT/NO81 sampled were still in a liquid state. These results are also compared to those obtained after four hours, but where the material has not been cured, in order to estimate how much of the deterioration is due to curing and how much is due to changes in the materials occurring whilst it was still in a liquid state. It must be noted that the cured samples, referred to as 'solid' in the table, are not retrained after curing. Instead, the solution found during training of these samples whilst they were still un-cured (liquid in the table) is reapplied to the now solid sample. This explains why the label 'N/A' is used for optimum training error in the case of the solid samples.

Results reported in Table 6.4 suggest that solutions evolved in un-cured SWCNT/NO81 are stable enough to be retained whilst the material is cured under UV light. In other words, the curing process does not appear to destroy completely the solutions evolved in the samples when they were in a liquid state. For both problems, the deterioration in error after curing, $\overline{\Phi}_e^v$ (solid), was higher than after leaving samples to rest for four hours, $\Phi_e^v(\mathbf{x}^* + 4h)$. This was especially significant since samples did not require 4hrs to cure. It is possible that the differences in verification errors are due to chemical changes in the epoxy resulting from the curing process.

The verification error in devices evolved to solve the V1C problem was on average 11.52%. This was 3.58% more than the original error. The loss of accuracy due to the curing process was more significant for the NLC dataset. In this case, the post-cure error was 16.18% higher than the liquid alternative, whilst the deterioration due to time was less that 2%.

The fact that the NLC dataset solutions appear to be more prone to change due to curing than those obtained for the V1C dataset was unexpected. If the same reasoning





used with SWCNT/LC samples was followed, the spread of data around the verification tests errors should be greater for the V1C problem, since it was suggested in Chapter 5 that more complex problems lead the DE algorithm to build more stable SWCNT structures in the material which would be less prone to variations in morphology induced by the algorithm-controlled electric field.

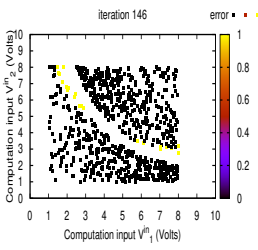
It must be noted that experiments involving the training of solidified samples resulted in training and verification errors significantly higher than those obtained with the same material in liquid state. There was therefore an advantage in using the liquid SWCNT/NO81 as a base for training, prior to curing.

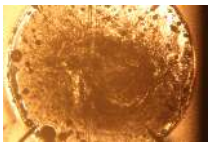
6.5 Summary of Results and Conclusions

Observations arising from the analysis of the EiM process on SWCNT/LC composites resulted in investigations into another type composite, where SWCNTs were combined with epoxies in liquid state. In this chapter, two SWCNT/epoxy (LP655 and NO81) composites were used to test the possibility of encapsulating devices. The aim was to allow algorithms to make use of the diversity in input/output behaviour provided by the SWCNT-based composites in liquid state for in-the-lab experiments *and* to allow the use of the resulting devices, in solid state, for out-of-the-lab applications.

The first composite tested, SWCNT/LP655, was subsequently abandoned as it was not possible to evolve it into an accurate classifier for the BCPs. In addition, at the SWCNT concentration selected for experiments, the composite viscosity was very high, and analysis of photographs taken during this training suggested little or no movement of nanotubes within the LP655 matrix.

The second composite tested, SWCNT/NO81, presented a much lower viscosity compared to SWCNT/LP655. In this case, it was possible to show that change in morphology within the material subjected to algorithm training was similar to that reported for the 0.05 wt % SWCNT/LC composite. Using the classical EiM framework and experimental implementation described in Chapter 3, it was possible to solve both a linear and non-linear separable classification problem with relative accuracy. Results were not as good as those obtained with SWCNT/LC samples, but the differences were not statistically significant from those obtained with either SWCNT/LC or SWCNT/PBMA.





The most interesting result reported in this chapter concerns the impact of the UV curing process on evolved solutions. It was possible to retain a level of classification accuracy after the SWCNT/NO81 composite had been cured under UV light. In other words, the material-based classifier evolved whilst in liquid state could be solidified, effectively resulting in an encapsulated device able to sustain physical tampering to some extent. The possibility to combine the benefits of both liquid and solid nanotube-based composites was suggested here.

However, results also suggested that, whilst the SWCNT/NO81 composite has the potential to solve computational problem through the EiM framework, a different implementation might be necessary to obtain more accurate and reproducible results. Indeed, the parameters used for the new SWCNT-based composites (maximum voltage level, number of decision variables, etc), had been optimised for the SWCNT/LC composite and might not be optimal for SWCNT/NO81. Further experiments involving changes in parameters, algorithm or EiM framework, have the potential to improve results and allow for more complex problems to be solved. In addition, curing the material resulted in an increase in verification error which was beyond that resulting from time-bound deterioration, suggesting a non-negligible impact of the curing process on the evolved solution. It would be interesting to see if a different implementation could result in structures presenting a strength similar to the one allowing SWCNT/LCs to keep memory of a past problem whilst having been subjected to a new sequence of training, and whether this strength would affect the ability of the SWCNT/NO81 to retain the accuracy obtained pre-curing.

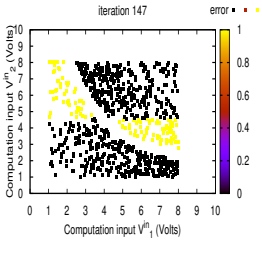
Bibliography

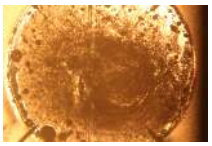
[1] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[2] S. L. Harding and J. F. Miller, "Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal," *Evolvable Systems: From Biology to Hardware*, pp. 155–164, 2005.

[3] J. Bird and E. Di Paolo, "Gordon pask and his maverick machines," in *The mechanical mind in history* (P. Husbands, O. Holland, and M. Wheeler, eds.), ch. 8, pp. 185–211, The MIT Press, 2008.

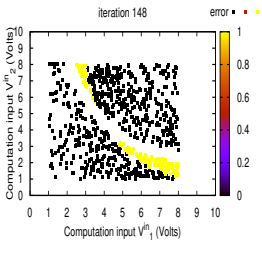
[4] D. Adhesives. https://www.delo-adhesives.com/fileadmin/datasheet/DELO20KATIOBOND_LP655_28TIDB-GB29.pdf, 2018.

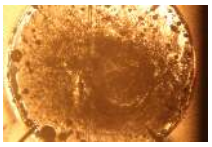




[5] N. P. Incorporated. https://www.norlandprod.com/msds/noa2081msd.html, 2017.

[6] Z. Wang, A. C. Bovik, H. R. Sheikh, E. P. Simoncelli, *et al.*, "Image quality assessment: from error visibility to structural similarity," *IEEE transactions on image processing*, vol. 13, no. 4, pp. 600–612, 2004.





Chapter 7

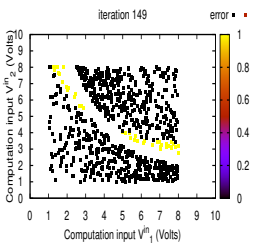
Solving Real-Life Problems with SWCNT/LC Composites

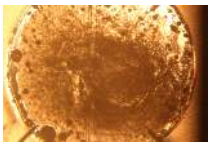
7.1 General Overview

Previous chapters reported solutions to synthetic binary classification problem (BCPs), obtained through training of single-walled carbon-nanotube (SWCNT)-based composites. The original motivation behind the use of these BCPs was to provide a proof-of-concept for a new material, SWCNT/ liquid crystal (LC).

Subsequent investigations with the same problems enabled the understanding of some of the complex mechanisms at play during the EiM process, such as the building of SWCNT structures resulting in a material memory and in the potential to learn from retraining. This understanding led to the suggestion of, and investigation into, SWCNT/epoxy composites for evolution in materio (EiM), which demonstrated the possibility to encapsulate solutions evolved in a material in liquid state for potential out-of-the-lab applications.

Results obtained with the SWCNT-based composites are difficult to compare with other techniques or even *in materio* implementations, since the BCPs were only used in one other study [1]. In addition, the BCPs are synthetic datasets which will not encompass all the difficulties that might be presented by real-life problems. Using benchmark problems, on the other hand, can give a flavour of how competitive a technique is compared to another, rather than the ability of the technique to find solutions to any





classification problem. The first aim of this chapter is therefore to determine whether liquid SWCNT/LC samples were capable of solving these more complex problems. The choice of relevant benchmark problems for testing new frameworks, algorithms or implementations has been the subject of much debate within the unconventional computing community [2], and the computing and electronics community at large [3]. In this context, the choice of problems was directed by the second aim of this chapter, which is to compare classifiers evolved in SWCNT/LC with those evolved in solid SWCNT/polymer and with *in silico* classifiers.

The Iris classification problem, based on the Iris dataset [4], has been commonly used to test SWCNT/poly(butyl meta acrylate) (PBMA) and SWCNT/ poly(methyl meta acrylate) (PMMA) composites as well as both the classical EiM framework [5–7] and reservoir computing in materio (RCiM) [8]. It is therefore the first problem investigated with SWCNT/ LCs, a differential evolution (DE) algorithm and the implementation of Chapter 3. The second set of investigations focused on the training of un-configured liquid materials for the solving of two medical problems, based on the mammographic mass dataset and bupa liver disorder referred to as MMC and bupa, respectively. These have not been used in other EiM-related investigations, but they provide a means of comparison with *in silico* classifiers obtained using the DE algorithm. The three datasets were retrieved from the UC Irvine (UCI) repository [9].

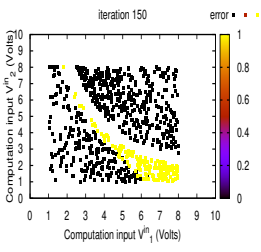
This Chapter is arranged in three sections, one for each problem. The Iris problem is presented first, along with a discussion of the results. The second and third sections are concerned with the MMC and bupa problems, respectively. Finally, a summary of discussion and results concludes this chapter.

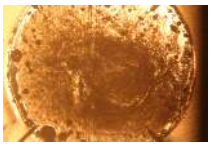
7.2 Iris Dataset

The Iris classification problem, which dataset was retrieved from the UCI repository [9] is commonly used to test new optimisation algorithms [4, 10–12] and as mentioned previously has been used as a benchmark in EiM investigations involving solid SWCNT-based composites.

The dataset consists in:

- 3 classes representing 3 types of Iris flowers (virginica, setosa and versicolor),





- 150 instances, 50 per class,
- and 4 attributes.

Unlike the type of classification problems investigated in the previous chapter, the Iris dataset represents a real-life classification problem, in the sense that instances from this dataset have been collected rather than generated for the purpose of an experiment. In addition, compared to all other problems investigated here and in previous chapters, the Iris datasets contains three classes, i.e. it is not binary.

The left hand side of figure 7.1 presents an Iris flower, along with the attributes which define its type: petal (length and width) and sepal (length and width). The dataset is represented in four dimensions in the right-hand side graph of Figure 7.1, with V_1 and V_2 corresponding to the petal measurements, whilst V_3 and V_4 correspond to the sepal length and width, respectively. One class, the iris Setosa, is fully separable from the other two. The Virginica and Versicolor are partially merged over two dimensions, i.e. attributes.

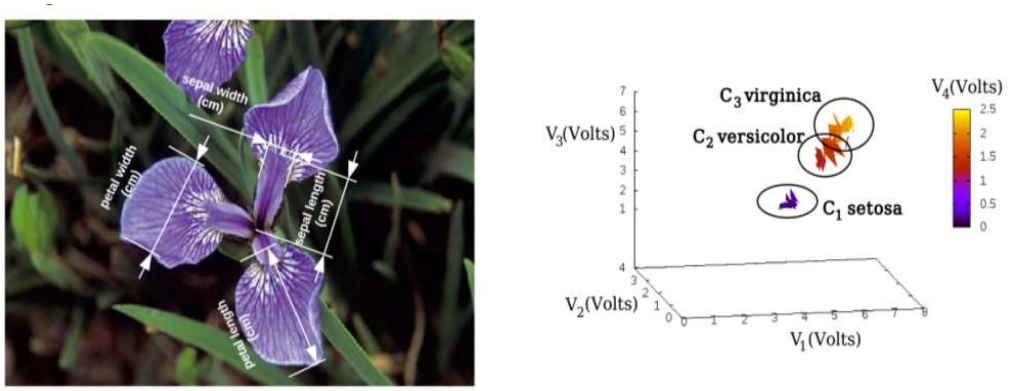
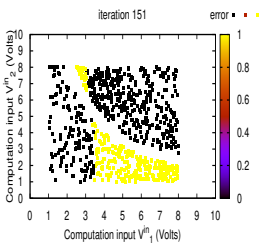


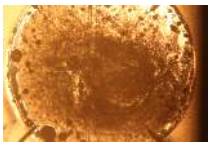
FIGURE 7.1: Photograph of an Iris versicolor, with the four attributes collected to build the Iris dataset illustrated four dimension.

7.2.1 Experimental Implementation

The implementation used to solve the Iris problem with SWCNT/LC differed from the one discussed in Chapter 4, since in the case of Iris, three rather than two classes were involved. Tests were undertaken to find a suitable function for the Iris problem, referred to as h^{iris} to be used to translate currents measured across the material into classes in the interpretation scheme ($S^{\mathcal{C}}$). Two different functions were tested.

In the first case, the evolvable motherboard (EM) used in experiments aiming at solving the BCPs, and presented in Chapter 3, Section 3.5, was left unchanged. The





function therefore had to assign data to three classes using two outputs. Preliminary results using this scheme proved globally poor, with around 33% error for both training and verification. This corresponds to a classifier that allows the correct classification of data belonging to the Setosa class, which is distinct from the other two, but is unable to distinguish between Versicolor and Virginica. In addition, a large number of iterations were required before the algorithm settled to the solution resulting in the correct classification of instances belonging to two out of the three classes.

The second scheme tested made use of three current outputs instead of two. The EM was modified to allow the three outputs to be collected across the material, making the material response $\mathbf{Y}(\mathbf{M})$:

$$\mathbf{Y}(\mathbf{M}) = \left[I_1(\mathbf{M})\ I_2(\mathbf{M})\ I_3(\mathbf{M})\right]^T \tag{7.1}$$

where $\mathbf{M}$ relates to the state of the material, but is not directly measurable.

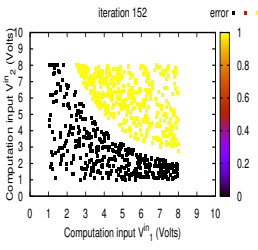
In this case, each output was compared to the other two. A good solution would be one that made output I_1, I_2 or I_3 highest in the presence of data belonging to the corresponding class C_1, C_2 or C_3. The full interpretation scheme, referred to as $S_{\mathcal{C}}^{iris}$ for the Iris dataset, was therefore as follows:

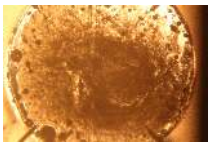
$$S_{\mathcal{C}}^{iris}(\mathbf{Y}) = \begin{cases} 1 & \text{if} \quad I_1 \geq I_2 \quad \text{and} \quad I_1 > I_3 \\ 2 & \text{if} \quad I_2 > I_1 \quad \text{and} \quad I_2 \geq I_3 \\ 3 & \text{if} \quad I_3 \geq I_1 \quad \text{and} \quad I_3 > I_2 \end{cases} \tag{7.2}$$

A list of important implementation parameters is presented in Table 7.1. The parameters of the DE algorithm used to evolve SWCNT/LC and SWCNT/PBMA samples into devices capable of solving the Iris problem are also reported in this table

TABLE 7.1: Experimental Parameters for the Iris Problem.

	Parameter	Value
DE	cross-over operator (CR)	0.7026
	differential weight (F)	0.814
Search	iteration size (Λ)	250-1000
	population size (N)	10
	$[V_{min}, V_{max}]$ (Volts)	[0, 4]
	$[p_{min}, p_{max}]$	[1, 182]
	$[V_{min}, V_{max}]$ (Volts)	[0, 4]
	$[R_{min}, R_{max}]$	[0.05, 15]
	$[p_{min}, p_{max}]$	[1, 182]
	scheme	h^{iris},





In order to train the material to solve the Iris problem, the dataset was split into a training and a verification set, both containing half of all instances, i.e 75 instances each. A minimum of five experiments was undertaken for each material.

7.2.2 Results

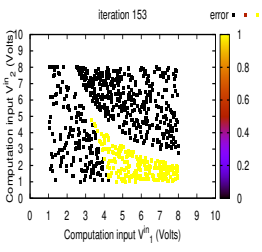
The performance of the EiM process in solving the Iris problem, whilst implemented in 0.05 wt % SWCNT/LC composites and trained using DE, is measured in terms of optimum training $\Phi_e^{t,*}$ error and average verification errors $\overline{\Phi}_e^v$. An estimate of the reproducibility of the process is provided by the variance, $\sigma(\Phi_e^v)$ in verification error across experiments.

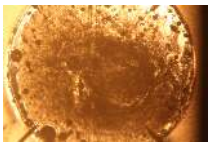
The three measures of performance are reported in Table 7.2. Where available, the same measures are reported for 0.71% SWCNT/poly (methyl metaacrylate) (PMMA) samples and 1 wt % SWCNT/PBMA samples. Results for these composites were obtained with the classical EiM [6, 7] and following the reservoir computing in materio (RCiM) framework [8]. In both cases, (1 + 4) evolutionary strategies (ES) was used as the algorithm and experiments were run in different versions of the EM. The last means of comparison was carthesian genetic programming (CGP) [13] mentioned in [8] and implemented on conventional hardware, i.e. *in silico*.

TABLE 7.2: Training and verification errors for the Iris problem

Material	framework	wt %	$\Phi_e^{t,*}(\%)$	$\overline{\Phi}_e^v(\%)$	$\sigma(\Phi_e^v)(\%)$
SWCNT/LC	EiM	0.05	13.00	17.5	4.77
SWCNT/PMMA	EiM [6]	0.71	16.3	22.9	N/A
	RCiM [8]		4.07	11.97	8.93
SWCNT/PBMA	EiM [7]	1	7.77	12.12	N/A
	RCiM [8]		2.86	8.1	5.48
in silico	CGP [8]	N/A	2.3	6.4	N/A

It must be noted that all the results with which the SWCNT/LC results are compared were obtained with different hardware platforms, algorithms and problem formulations, in addition to being obtained with different SWCNT-based composites. It is therefore difficult to analyse whether the difference in results was due to the material only.





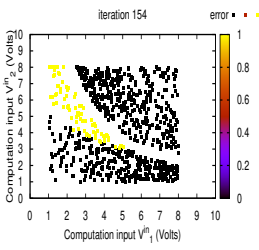
Errors obtained with SWCNT/LC samples were 13% and 17.5% for training and verification respectively. This suggests that DE-controlled evolution has created a device with the ability to solve more than linear classification problems. If linear classification was the only possible task that the evolved SWCNT/LC samples could perform, the verification error would have been around 33%, which is the lowest possible error that a linear classifier with a single threshold could achieve on the Iris dataset.

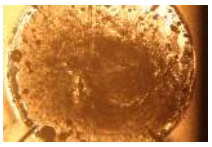
Overall, results reported in the table suggest that the framework used to train materials has a greater effect on the training and verification errors than the material used. The level of these errors obtained with the SWCNT/LC composite were comparable to both SWCNT/PMMA (EiM), SWCNT/PBMA (EiM), being slightly lower than the first and higher than the second. However the SWCNT/LC errors were higher than the training and verification errors obtained through the RCiM framework, irrespective of the material, and higher than CGP *in silico*. It would be interesting to investigate whether RCiM would benefit, in the sense of more accurate results, faster convergence etc, from using the dynamic SWCNT/LC composites rather than solid SWCNT/PBMA.

The spread in results across experiments is the only performance measure suggesting an advantage in using SWCNT/LC, in the current experimental implementation, over the other material and framework alternatives. The standard deviation across the verification errors obtained with SWCNT/LC samples was lower than that obtained with the RCiM framework in both SWCNT/PMMA and SWCNT/PBMA.

This is coherent with the hypothesis formulated in [14] and discussed in [8], i.e. the finding of a solution in solid SWCNT-based composites is dependent on the SWCNT structures present during the material's solidification process. This is a constraint that is not present in SWCNT/LCs. Results presented here suggest that the latter's flexible state, one that can be transformed during training, allows a level of reproducibility in results which is higher than in the solid samples.

The Iris problem is an important benchmark problem in machine learning and related research. It was investigated mainly because it provided a mean of comparison with other EiM implementations and SWCNT-based materials. However, as argued in [3], the fact that a technique can solve this problem does not mean that it will be able to do well at other, more complex ones. In order to provide more ground for the analysis of SWCNT/LC composite performance in dealing with real-life problems, the next





two sections investigate this material's ability to solve problems based on two medical datasets.

7.3 Mammographic Mass Dataset

The mammographic mass problem (MMC) is characterised by $n_1 = 4$ features of a breast mass, which is deemed either carcinogenic or benign. Three features follow a discrete qualitative marking scheme (mass margin, shape and density), whereas the fourth (patient age) is a continuous quantity.

The dataset, retrieved from the UCI repository [9], has a total of 961 instances, split in a 54:46 ratio between the two classes. In the experiments undertaken here, the 130 instances presenting one or more missing attribute have been removed from the dataset. This is also the case in the paper presenting the results with which the performance of the evolved SWCNT/LC classifiers is compared [15]. When the instances with missing attributes are removed, the split between the two classes becomes close to a 50:50 ratio, which is conserved in the training and verification datasets.

Figure 7.2 presents mammograms with no mass at all, benign cyst and carcinogenic mass. The different attributes of the MMC dataset are represented in three dimensions over two graphs below the mammograms. The two classes correspond to no mass and carcinogenic mass.

The same two complexity metrics used for the synthetic BCPs are used for ranking the difficulty of each of the MMC problem: the Fisher criterion and the volume of overlap [17].

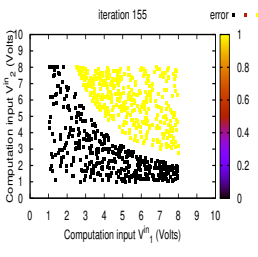
The Fisher criterion for an arbitrary BCP, $F_{1,BCP}$, is defined as

$$F_{1,BCP} = \max_{j=1,\ldots,n_1} \left\{ \frac{\left(\mu_{1,j} - \mu_{2,j}\right)^2}{\sigma_{1,j}^2 + \sigma_{2,j}^2} \right\} \tag{7.3}$$

where $\mu_{i,j}$ and $\sigma_{i,j}$ is the mean and the standard deviation of feature j for class i, respectively. The higher the value of $F_{1,BCP}$ the less complex is the problem and therefore should be easier to solve.

The volume of overlap metric for an arbitrary BCP, $F_{2,BCP}$, is defined as

$$F_{2,BCP} = \prod_{j=1}^{n_1} \frac{\min\left\{U_{1,j}, U_{2,j}\right\} - \max\left\{L_{1,j}, L_{2,j}\right\}}{\max\left\{U_{1,j}, U_{2,j}\right\} - \min\left\{L_{1,j}, L_{2,j}\right\}} \tag{7.4}$$



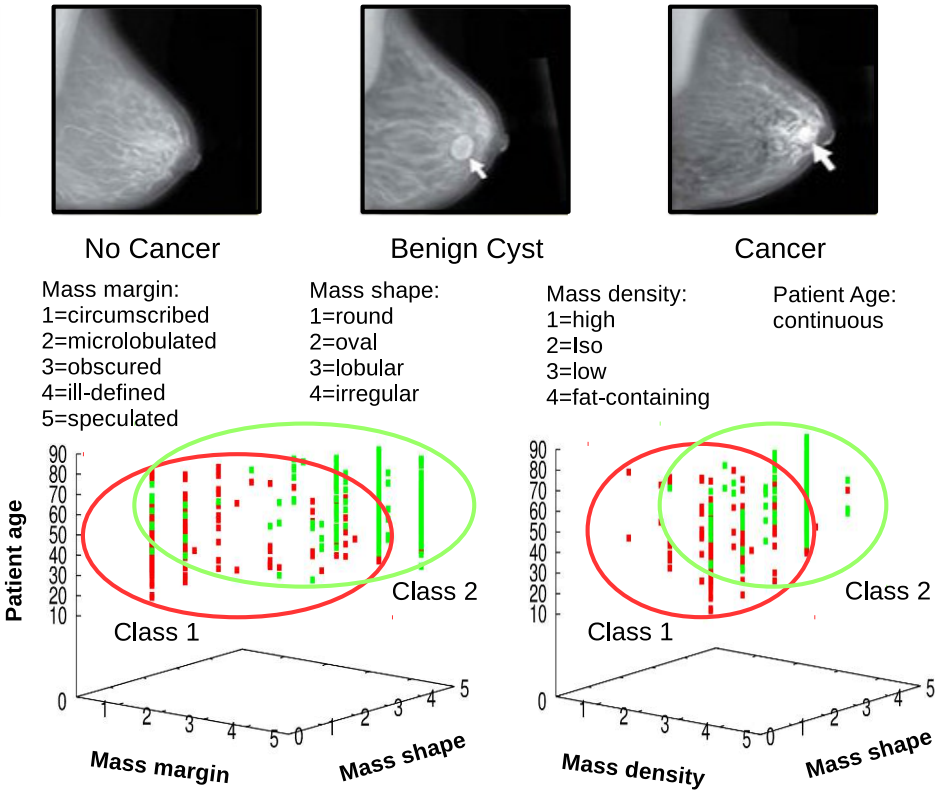


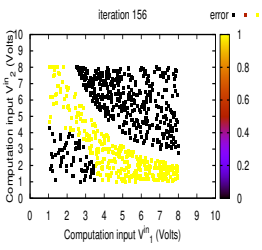
FIGURE 7.2: Mammographic mass (photo taken from [16]).

where

$$U_{1,j} = \max\left\{V_j^{\mathcal{C}} : \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_1\right\}, U_{2,j} = \max\left\{V_j^{\mathcal{C}} : \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_2\right\}$$
$$L_{1,j} = \min\left\{V_j^{\mathcal{C}} : \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_1\right\}, L_{2,j} = \min\left\{V_j^{\mathcal{C}} : \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_2\right\}. \tag{7.5}$$

The multiplicative form of $F_{2,BCP}$ indicates that if the two classes are separable in just one feature, it will take a zero value. The higher the value of $F_{2,BCP}$ the more complex the problem is and therefore it should be more difficult to solve.

Table 7.3 summarises the parameters describing the four BCP previously considered, along with the new problem. This provides a better idea of the relative complexity of each material, and specifically of the MMC problem as compared to the V1C, SC, MC, and NLC problems previously solved with SWCNT/LC composites. For each problem, the number of features n_1 is given along with the total number of points in the training and the verification datasets, i.e $K_t + K_v$; the problems are placed in ascending order of complexity, as indicated by the values of $F_{1,.}$ and $F_{2,.}$ for each of them.



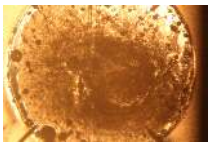


TABLE 7.3: Parameters of the synthetic BCPs and the medical dataset, MMC.

Problem	n_1	$K_t + K_v$	$F_{1,.}$	$F_{2,.}$
V1C	2	4800	2.198	0
SC	2	4800	2.075	0
MC	2	4800	1.908	6.6
NLC	2	4800	0.962	0
MMC	**4**	**831**	**0.809**	**1.4**

7.3.1 Experimental implementation

A simple linear transformation is used to scale all computational inputs to the range of $[0, 4]$ Volts, irrespective of whether they are continuous or not. Hence,

$$\mathcal{A} = [0, 4]^4 \tag{7.6}$$

with $\mathcal{D} = \{1, 2\}$, since the problem is binary.

The implementation parameters are presented in table 7.4, along with the parameters of the DE algorithm used to evolve the samples of SWCNT/LC composites and microtubule solutions into solving the MMC dataset. A minimum of five experiments was undertaken for each material and material concentration.

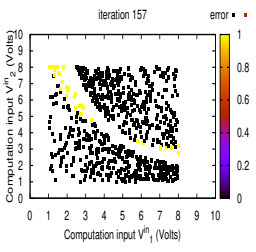
TABLE 7.4: Experimental Parameters for the MMC problem.

	Parameter	Value
DE	cross-over operator (CR)	0.7026
	differential weight (F)	0.814
Search	iteration size (Λ)	150-300
	population size (N)	10
	$[V_{min}, V_{max}]$ (Volts)	[0, 4]
	$[R_{min}, R_{max}]$	[0.05, 15]
	$[p_{min}, p_{max}]$	[1, 182]
	scheme	$h^{(2)}$

7.3.2 Results

Effect of concentration

Table 7.5 presents experimental results from investigations related to the range of concentrations, [0.015-0.1] wt % SWCNT/LC. If un-configured, a sample classifies the problem's data randomly, resulting in 50% error. Φ_e^* and $\Phi_{e,v}^*$ are the optimum error obtained during training and verification of the sample respectively. Ten tests are performed during the verification procedure. $\overline{\Phi}_{e,v}$ is the average and the σ the variance over these tests.



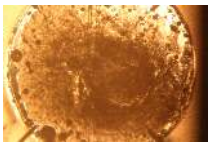


TABLE 7.5: Performance of evolved SWCNT/LC composite, at different concentrations, for the MMC problem

SWCNT wt %	$\Phi_e^*(\%)$	$\Phi_{e,v}^*(\%)$	$\overline{\Phi}_{e,v}$	$\sigma(\%)$
0.015	24.5	49.49	49.49	0.000
0.05	20.5	18.85	20.507	0.5133
0.1	20.3	19.65	20.84	0.5638

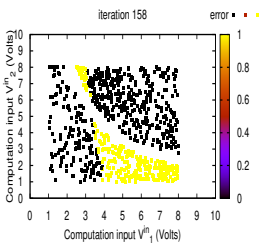
It can be observed that materials of 0.015 wt % are unable to generalise well, that is, classify correctly the previously unseen verification data. On the other hand, increasing the SWCNT concentration to 0.05 wt % and 0.1 wt % yields results demonstrating improvement from the undiscerning state (for which error is 50%), good generalisation and good reproducibility, illustrated by the average verification error $\overline{\Phi}_e^v$ and the standard deviation $\sigma(\Phi_e^v)$, respectively.

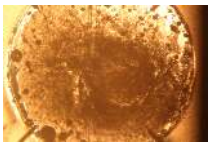
Comparison with *in silico* classifiers

In order to assess the solutions currently obtained *in materio*, results to MMC problem, averaged over five tests, are presented in Table 7.6. They are compared with those achieved over a range of neural network (NN) implementations running on conventional computers, i.e. *in silico*. Results for the MMC problem are reported in [15], where a dentrite morphological NN (DMNN) is trained using DE. Finally, the SWCNT/LC results are compared with those obtained from a medical survey [18] where human accuracy on an equivalent to the MMC problem is investigated. Only two extremes are reported here, results for fellowship trained radiologists, as defined in [18], who are very good at discerning correctly carcenogenic masses from mammograms and non-fellowship trained radiologists. The radiologists only had access to mammograms, however, the specific mammograms used in this study were different than those used in the UCI repository. In the table, the diagnosis of the first are reported as best verification error ($\Phi_v^{v,*} = 12.00\%$

TABLE 7.6: Different implementation performance in solving the MMC problem.

MMC	Material	$\Phi_e^*(\%)$	$\Phi_e^{*,v}(\%)$	$\overline{\Phi}_e^v$ (%)	$\sigma(\Phi_e^v)(\%)$
EiM, DE	SWCNT/LC	20.5	18.85	20.51	11.097
DMNN, DE [15]	*in silico*	15.8	N/A	10.40	N/A
Human learning [18]	*brain*	N/A	12.00	17.00	N/A





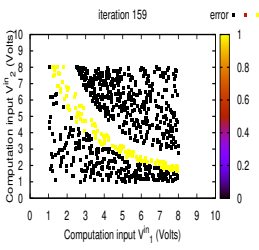
error) and the diagnosis of the non-fellowship trained radiologists are reported as average verification error $\overline{\Phi}_e^{\upsilon} = 17.00\%$ error. The difference between the two radiologist diagnosis is detailed in [18]. It can be seen that despite the problem's complexity, the SWCNT/LC material can be brought in a state where it is able to classify at best 18.85% and on average 20.51% of the 631 instances contained in the MMC verification dataset. This is less than half the error that would be obtained if the material was randomly assigning data to one class or another.

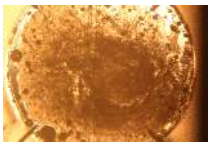
As mentioned previously, the MMC dataset was split into $K_t = 200$ and $K_v = 631$. Results for DE-trained DMNN were obtained using a ratio of training to verification instances nearly inverted, i.e. $K_t = 664$ and $K_v = 167$. However, the evolved SWCNT/LC blend is able to produce a training error which is 'only' 4.7% higher, whilst the average verification error is 10.11% superior to the DE-trained DMNN. The best and average *in materio* solutions compare better with the diagnosis of non-fellowship trained radiologists ($\overline{\Phi}_e^{\upsilon}$), with an error that is respectively 1.85% and 3.51% higher. The fellowship trained radiologists ($\Phi_e^{*,\upsilon}$), on the other hand are more accurate in their diagnosis than both their non-fellowship trained counterpart and the SWCNT/LC.

Due to hardware constraints, DE was implemented with a population of ten individuals, and with ten decision variables, as presented in section 7.3.1. This is lower than the population size and parameter number used in [15], and in most DE implementations. Nonetheless, in no more than four hundred iterations, it was possible to produce an evolved material able to classify a number of instances from the two datasets without the metal-oxide-semiconductor-field-effect-transistors (MOSFET) components crucial to *in silico* implementations.

7.4 Bupa Liver Disorder

The Bupa liver disorder (bupa) problem is characterised by data with $n_1 = 6$ features describing a patient's liver condition and is often used as a medical binary classification problem to test and compare machine learning methods and algorithms. An EiM approach to the solving of this problem was investigated by the author of this thesis. Results obtained with SWCNT/LC samples were published in [20]. However following the publication of [21] by McDermott and Forsyth (the latter provided the dataset for the bupa liver disorder problem), it has appeared that most published research involving





bupa had misused the dataset from the moment it was deposited on the UCI repository. It is also the case in the work presented in [20], and in the two publications [15, 19] reporting the results with which the evolved SWCNT/LC classifiers are compared.

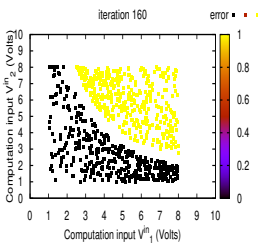
The dataset of the Bupa liver disorder problem, as published in the UCI repository, does not come with classes to which the data should belong. It is therefore not suitable for supervised learning. The attribute commonly considered as the dataset's classes is instead an indicator of how to split the dataset between training and verification. As a result, the comparisons presented here do not represent the different classifier's ability or inability to discriminate between healthy and cirrhotic liver.

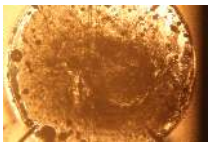
7.5 Summary of Results and Conclusions

This chapter investigated three real-life classification problems retrieved from the UCI repository [9]. Contrary to the synthetic BCPs used previously, the three real-life problem's datasets contain instances that are spread across more than two dimension,s with non-linear boundaries and overlapping areas. In the case of the medical datasets, they combine continuous and discrete data.

Investigations into the solving of these problems by means of classical EiM, DE and SWCNT/LC composites provided comparison with more conventional classification techniques, but also with other implementations of EiM and different materials.

- **Iris**: results obtained for the Iris problem suggested that SWCNT/LC composites provided an advantage over other SWCNT-based materials used for *in materio* computation (including both classical EiM and RCiM): classification error levels were more reproducible in SWCNT/LC samples than in the solid SWCNT-based samples, irrespective of the framework employed to find the solution. However, it was also observed that for this problem, the accuracy provided by the SWCNT/LC was not comparable with that obtained with solid SWCNT/PMMA and SWCNT/PBMA composites trained using the RCiM framework and a (1+4) evolutionary strategy algorithm mentioned in Chapter 3. It was noted that a more elaborate analysis of results could be provided if difference between the experiments were limited to different material. Instead, results presented were obtained in experiments where all implementation parameters differed from those used when training the SWCNT/LC composite.





- **Medical datasets:** The mammographic mass and bupa liver disorder problems. Results for the MMC dataset were higher than those obtained by neural network implementations [15, 18] and human diagnostic. The interpretation scheme and objective function used in the implementation was the same used to solve the less complex synthetic BCPs and it is very simple. For example, the difference between true and false positives, an important parameter in medical applications, was not used in the problem formulation.

Bibliography

[1] F. Qaiser, *Training Single Walled Carbon Nanotube Based Materials to Perform Computation (PhD Thesis)*. 2018.

[2] E. Blakey, "Unconventional computers and unconventional complexity measures," in *Advances in Unconventional Computing*, pp. 165–182, Springer, 2017.

[3] Various, "Irds^{TM} 2016 edition, applications benchmarking white paper," 2016.

[4] R. A. Fisher, "The use of multpitle measurements in taxonomic problems," *Annals of Eugenics*, vol. 7, no. 2, pp. 179–188, 1936.

[5] K. D. Clegg, J. F. Miller, M. K. Massey, and M. C. Petty, "Practical issues for configuring carbon nanotube composite materials for computation," in *2014 IEEE International Conference on Evolvable Systems*, pp. 61–68, IEEE, 2014.

[6] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.

[7] M. Mohid and J. F. Miller, "Evolving solutions to computational problems using carbon nanotubes.," *International journal of unconventional computing*, vol. 11, 2015.

[8] M. Dale, *Reservoir Computing In-Materio (PhD Thesis)*. 2018.

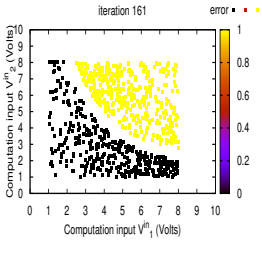
[9] D. Dua and C. Graff, "UCI machine learning repository," 2017.

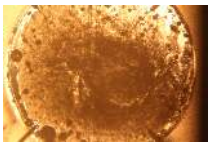
[10] M. Schmuker, T. Pfeil, and M. P. Nawrot, "A neuromorphic network for generic multivariate data classification," *Proceedings of the National Academy of Sciences*, vol. 111, no. 6, pp. 2081–2086, 2014.

[11] B. Dennis and S. Muthukrishnan, "Agfs: Adaptive genetic fuzzy system for medical data classification," *Applied Soft Computing*, vol. 25, pp. 242–252, 2014.

[12] P. Melin, F. Olivas, O. Castillo, F. Valdez, J. Soria, and M. Valdez, "Optimal design of fuzzy classification systems using pso with dynamic parameter adaptation through fuzzy logic," *Expert Systems with Applications*, vol. 40, no. 8, pp. 3196–3206, 2013.

[13] J. F. Miller and S. L. Harding, "Cartesian genetic programming," in *Proceedings of the 10th annual conference companion on Genetic and evolutionary computation*, pp. 2701–2726, ACM, 2008.





[14] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.

[15] F. Arce, E. Zamora, H. Sossa, and R. Barrón, "Dendrite morphological neural networks trained by differential evolution," in *Computational Intelligence (SSCI), 2016 IEEE Symposium Series on*, pp. 1–8, IEEE, 2016.

[16] J. J. Heidelbaugh and M. Bruderly, "Cirrhosis and chronic liver failure: part i. diagnosis and evaluation," *Am Fam Physician*, vol. 74, no. 5, pp. 756–62, 2006.

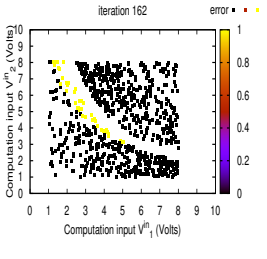
[17] T. K. Ho and M. Basu, "Complexity measures of supervised classification problems," *IEEE transactions on pattern analysis and machine intelligence*, vol. 24, no. 3, pp. 289–300, 2002.

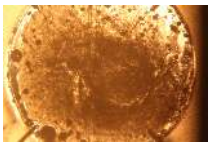
[18] J. G. Elmore, S. L. Jackson, L. Abraham, D. L. Miglioretti, P. A. Carney, B. M. Geller, B. C. Yankaskas, K. Kerlikowske, T. Onega, R. D. Rosenberg, *et al.*, "Variability in interpretive performance at screening mammography and radiologists' characteristics associated with accuracy 1," *Radiology*, vol. 253, no. 3, pp. 641–651, 2009.

[19] P. Jeatrakul and K. W. Wong, "Comparing the performance of different neural networks for binary classification problems," in *Natural Language Processing, 2009. SNLP'09. Eighth International Symposium on*, pp. 111–115, IEEE, 2009.

[20] E. Vissol-Gaudin, A. Kotsialos, C. Groves, C. Pearson, D. A. Zeze, and M. C. Petty, "Computing based on material training: Application to binary classification problems," *2017 IEEE International Conference on Rebooting Computing (ICRC)*, pp. 1–8, 2017.

[21] J. McDermott and R. S. Forsyth, "Diagnosing a disorder in a classification benchmark," *Pattern Recognition Letters*, vol. 73, pp. 41–43, 2016.





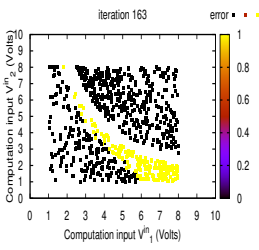
Chapter 8

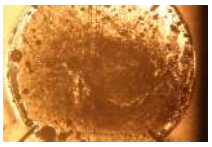
Alternative Evolutionary Substrates

8.1 General Overview

The field of evolution in materio (EiM) has been developed within the context of unconventional computing (UC), and with the aim of testing the ability of algorithms to configure different types of materials into devices able to perform a computation. Since EiM was first proposed in [1], three types of material have been investigated experimentally: liquid crystal displays (LCDs), gold nanoparticle dispersions [2] and single-walled-carbon-nanotube (SWCNT)-based composites. Investigations relating to the latter are reported in Chapters 4 - 7.

SWCNT-based composites were chosen mainly for their electrical properties, and the current interest for nanotube-based technology. This interest has been motivated by carbon nanotube properties such as high electron mobility, variation in band structure (metallic/semi-conducting) and large anisotropy. Composites of SWCNTs dispersed in poly vinyl acrylate (PVA), poly(methyl meta acrylate) (PMMA) and poly(butyl meta acrylate) (PBMA) were proposed and developed within the context of the Nascence project, a collaborative effort to further the field of EiM [3–5]. The SWCNT/PMMA and SWCNT/PBMA, showed the most potential for being evolved into a variety of computing devices using EiM [6–15] and reservoir computing in materio (RCiM) [16, 17] described in Chapter 1. Other candidate materials, and material properties, have also been suggested [18, 19], and in some cases investigated [20, 21]. However, the main



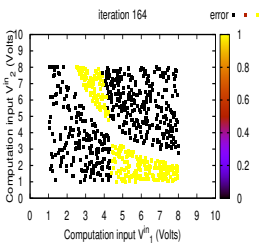


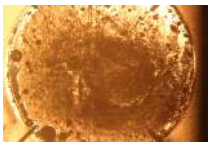
other type of candidate material investigated, gold nanoparticle particle suspension, is relatively similar to SWCNT-based composite, and except for [2], it was tested *in silico* [20, 21], i.e. using models, rather than *in materio*. On the other hand, materials such as bacterial consortia, [22], slime moulds [23] and memristors have been successfully evolved into devices capable of solving computational problems. Investigations related to these materials did not follow the EiM framework, but the principles were similar, suggesting the possibility for materials from biological origins, or with biological-like characteristics to be used in the field of EiM. This motivates the investigations reported in this chapter, which is concerned with evolving two new materials: microtubules and memristors. The motivations behind the two specific materials are discussed, along with the way they have been produced, their electrical characteristics, and their computational response.

8.2 Electro-Biological EiM Processor: Classifying Data with Microtubules

8.2.1 Motivations

Microtubules (mTs) are a new type of material considered in EiM research. These tubular structures are protein lattices present in eukaryiotic cells [24]. MTs, schematically illustrated in Figure 8.1, are of approximately 25 nm diameter. Their length is dynamic, but can reach up to a few microns, depending on the specialisation of the cell within which they are found [25, 26]. MTs are an important element of the cytoskeleton and can have different functions such as helping with cell division, as tracks for protein transport or as supporting structures in neuronal cells. Another function, as quantum information processors involved in memory and consciousness, has been proposed by Penrose and Hameroff [27–29]. The biological feasibility of this function has been the subject of controversy, especially in view of recent advances in the understanding of mT formation and characteristics [30–32]. However, whether or not this model can be proven, the importance of these structures in information processing within the brain, their electrical properties [33–35], and the fact that they are simpler than the highly evolved slime moulds or bacterial consortia often used as biological media in UC, makes them an interesting and potentially suitable subject of study within EiM research. It must be noted





that very recent work in the field of UC has been concerned with mTs-based systems and their potential for the development of analogue computing systems [36].

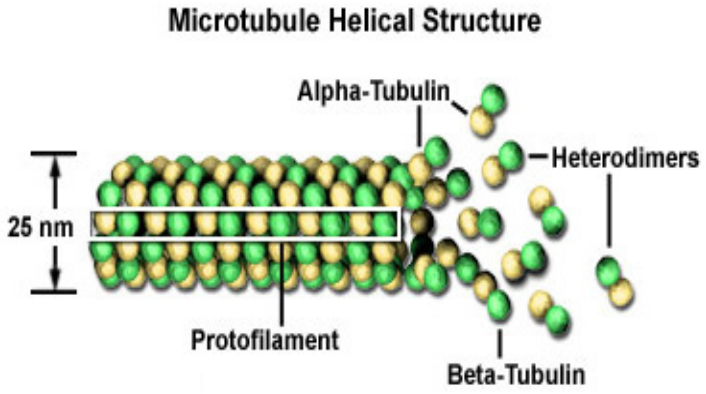
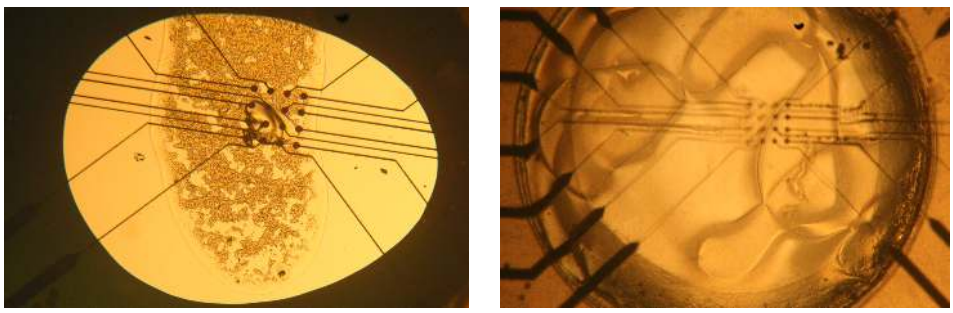


FIGURE 8.1: Schematics of a microtubule structure from [37].

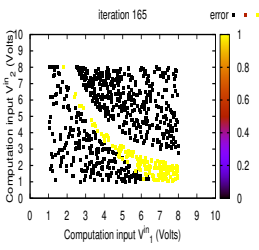
Sample Preparation

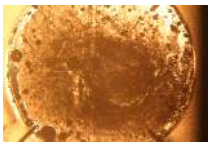
The biological samples used in this work were provided by Prof. Horacio Cantiello [34], formerly of the University of Buenos Aires and now director of the Instituto Multidisciplinario de Salud, Tecnología y Desarrollo (IMSaTeD) in Argentina. They were extracted from bovine brains and deposited on the 16-terminal electrode arrays in dry form. Three amounts of mTs were deposited: 4, 8 and 12 microlitres (μL). Tests were performed on dry and rehydrated samples. Water was used to rehydrate the mT films. Rehydrated samples have concentrations of 17.23, 34.26 and 51.39 % of mTs, respectively. It can be observed from Figure 8.2(a) that the dry mT deposition covers all the electrode terminals, but not the overall surface encompassed by the washer. Figure 8.2(b) shows that the solution obtained when rehydrating the sample can cover the whole washer circumference.



(a) (b)

Figure 8.2: Microtubules (a) dry and (b) rehydrated films.





Electrical characteristics of microtubules

Single mTs subjected to electrical stimulation have been observed to amplify signals, suggesting that they behave as biomolecular transistors [33]. Similar observations were reported for mT sheets [34] and mT bundles [35], which, in addition, tend to spontaneously generate electrical oscillations. The current / voltage characteristics reported in these investigations show a linear relationship when voltages between -40 mV to 40 mV are applied, with an increasing voltage step of 1 mV. However, the mechanism behind the mTs and mT-based structure's electrical activity is not yet well understood, and the range of voltages used in the I/V characterisation [34, 35] is limited, as compared to the range needed in EiM experiments, as implemented in the work undertaken here. The mT samples described in the previous section were therefore subjected to a series of voltage sweeps and their current outputs collected across the micro-electrode array.

The resulting graph, presenting the I/V curve obtained from voltage sweeps between 0 V to 10 V are presented in Figure 8.3. It can be observed that for the higher mT concentrations, i.e. 8μL and 12μL, the maximum output current was in the range $10^{-1}\mu$A, similar to low concentration SWCNT-based composites, and high enough to be recorded by the EiM hardware. Current outputs of the 4μL are much lower, and was closer to the values produced by ≤ 0.5 wt $\%$ SWCNT/PBMA. All samples presented a linear I/V, with steep increases in current outputs, especially for the two higher concentrations, which

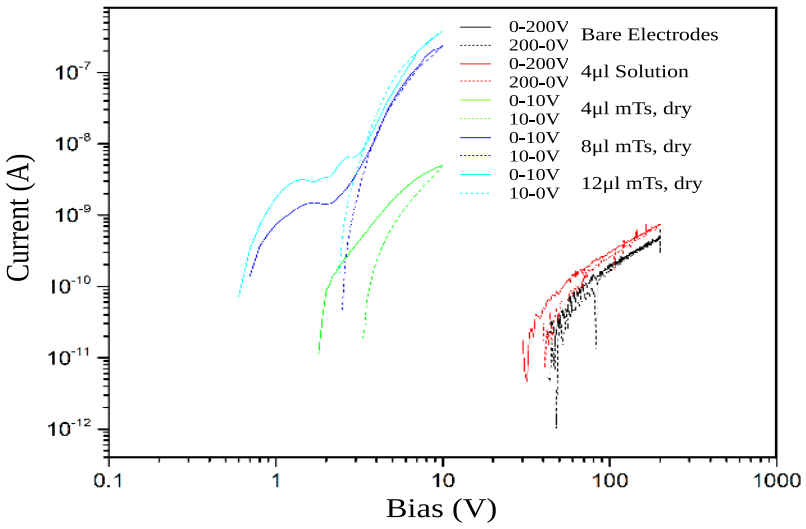
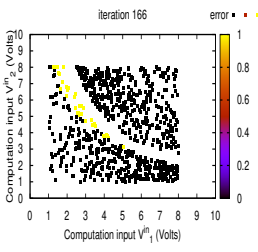
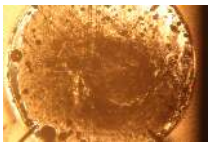


Figure 8.3: I/V characteristics of bare electrodes, $4\mu L$ solution and $4\mu L$, $8\mu L$ and $12\mu L$ dry microtubule samples deposited on the micro-electrode arrays.





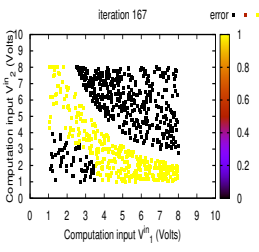
follows the observations reported in [34, 35]. The presence of an anticlockwise hysteresis, most important before 2V and suggesting a charge trapping mechanism within the sample must be noted. It is consistent with the observation reported in [36], despite the difference in the type of mT used. The I/V of rehydrated samples were not measured, as the solution tends to evaporate throughout the experimental time. The current output levels for the two higher mT concentrations satisfy the most necessary requirement of materials for EiM.

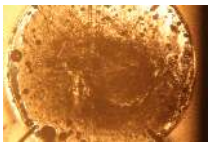
8.2.2 Experimental Implementation

The rehydrated mTs deposited upon the micro-electrode arrays were trained first to solve the SC problem, as it is the simplest problem for the implementation of EiM developed here. In the second set of experiments, the material was trained to solve the mammographic mass (MMC) problem, defined in Chapter 7, Section7.3. The aim was to evaluate this material's ability to solve a more complex computational problem than the SC problem. In addition, the dataset defining the MMC problem is available online [38] rather than custom-build for the purpose of these experiments, allowing comparisons between the evolved mT processor and other classifiers produced using machine learning. The SC and MMC problems are therefore used as a proof-of-concept, and a benchmark, respectively, for the analysis of the EiM-trained electro-biological classifier.

Depending on the dataset, two different functions were used in the interpretation scheme that transforms the output currents measured across the mTs into an error. When training the material to solve the SC problem, i.e. with the SC training and verification datasets, the function $h^{(1)}$ defined in Eq. (4.7) was implemented. On the other hand, the function $h^{(2)}$ defined in Eq. (4.12) was used for the more complex MMC problem. This difference in the problem formulation between the two sets of experiments is motivated by the discussion and results reported in Chapter 4 with the SWCNT/LC composites. It was observed that $h^{(1)}$ was sufficient to solve the SC problem, but that solutions producing low errors could not be found for the more complex problems using this equation. Implementing $h^{(2)}$ yielded better results. These discussions are based on different material than the one investigated here.

The mTs were trained using the differential evolution algorithm (DE). The algorithm's and search parameters are reproduced in Table 8.1 for the sake of clarity. The





parameters do not differ from those used in the SWCNT/LC experiments undertaken earlier with the SC and MMC problem.

TABLE 8.1: Experimental Parameters for the MMC problem.

	Parameter	Value
DE	cross-over operator (CR)	0.7026
	differential weight (F)	0.814
Search	iteration size (Λ)	150-300
	population size (N)	10
	$[V_{min}, V_{max}]$ (Volts)	[0, 4]
	$[R_{min}, R_{max}]$	[0.05, 15]
	$[p_{min}, p_{max}]$	[1, 182]
	scheme (SC)	$h^{(1)}$
	scheme (MMC)	$h^{(2)}$

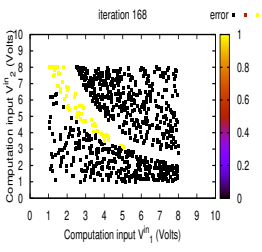
8.2.3 Comparison with SWCNT/LC and *In Silico* Classifiers

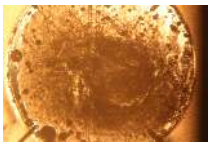
The optimum training and verification errors obtained with the microtubules, and averaged over experiments, are reported in Table 8.2. The best verification error obtained across these experiments is also reported, along with a measure of the spread of reproducibility illustrated by the standard deviation in verification error. The results obtained with microtubules are compared to those obtained with the SWCNT/LC composites.

Results obtained with the SC dataset suggest that the material has the capacity to be evolved to a better extent than control samples such as LCs and empty electrodes (for which the best and average verification errors are always around 50%). The best training and the best verification errors, averaged across experiments with mTs are 23.9% and 26.278%, respectively. This suggests a potential for the solutions evolved in mTs to generalise well to unseen data, since the difference between best training and best

TABLE 8.2: Comparing evolved classifier performance between SWCNT/LC and microtubules

Problem	Material	$\Phi_e^{t,*}(\%)$	$\Phi_e^{v,*}(\%)$	$\overline{\Phi}_e^v$	$\sigma_{\Phi_e^v}(\%)$
SC	SWCNT/LC	0.255	0.013	1.534	1.522
	mT	23.900	26.278	44.178	10.142
MMC	SWCNT/LC	20.51	18.859	31.067	11.785
	mT	38.033	47.543	47.881	0.362





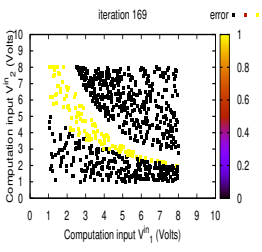
verification errors was below 3% However, the quality of the solution varied across experiments, as illustrated by the 10.142% standard deviation, and as a result, the average error obtained with the SC verification dataset was 44.178%.

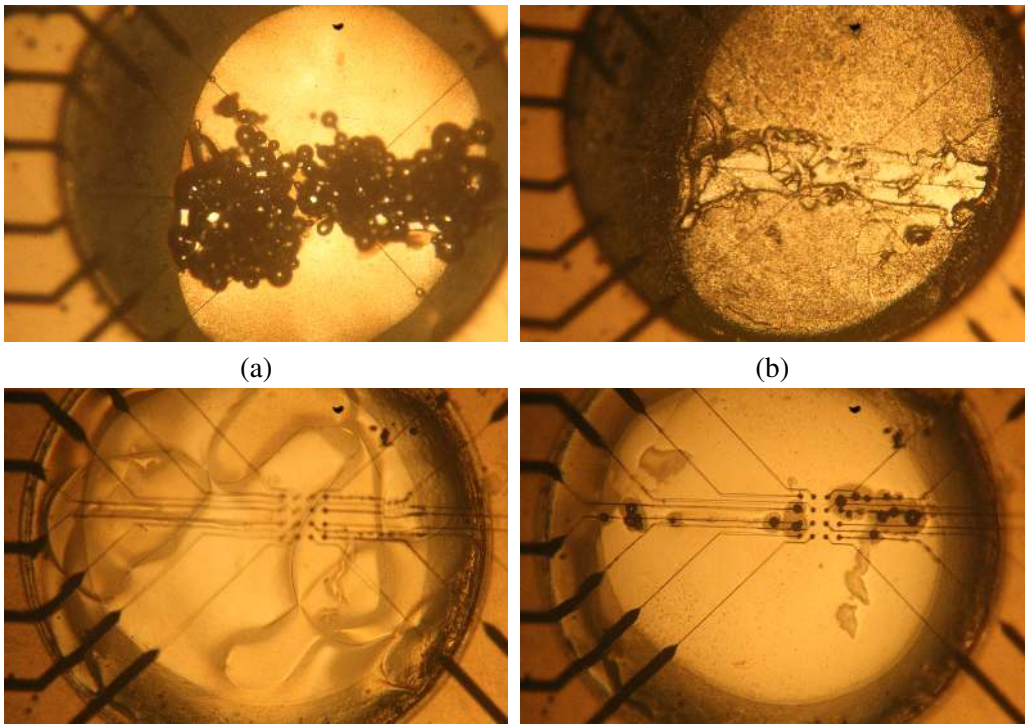
From the second set of results reported in Table 8.2, it can be observed that it was not possible to evolve the microtubules to solve the MMC dataset. Both training and verification errors obtained with this material ($\Phi_e^{t,*} = 38.033\%$ and $\overline{\Phi}_e^{v} = 47.881\%$) were significantly higher than those obtained in SWCNT/LC ($\Phi_e^{t,*} = 20.51\%$, $\overline{\Phi}_e^{v} = 31.067\%$).

A number of tests were undertaken in an attempt to improve the performance of the material, including changes in maximum configuration voltage, function used in the interpretation scheme or microtubule concentration, with no difference in results.

It is noted that the water used to rehydrate the samples before training tended to evaporate before the final iteration was achieved. As a result, the samples' electrical properties changed from low current outputs at the start of training, to negligible towards the end. A method was devised to prevent water evaporation during training. A transparent cap was fixed above the material and sealed using the same two part epoxy used to fix the washers on the micro-electrode array, as described in Chapter 2. Using this method, it was possible to keep the mTs rehydrated, whilst also allowing the capture of photographs monitoring the changes in material morphology, if any, induced by the training process. However, instead of helping in the finding of solutions to the classification problems, the addition of the cap over the material resulted in the deterioration of the material layer during the algorithm's search. The water tended to boil under the application of the configuration and computation voltages, a layer of bubbles formed along the electrodes covered by the material. Once experiments ended and the cap was removed, allowing the water to evaporate, it was observed that the layer of mTs situated above the electrodes, i.e. where the bubbles had appeared, was peeling off the microscope slide. The deterioration of the material due to training is illustrated in Figure 8.4.

It is possible that another liquid, such as liquid crystals, could be used instead of water to disperse the mTs. This combination was tested experimentally against the SC dataset. Figure 8.4 (c) presents the resulting sample. No evaporation of the LCs was observed, and the mT layer remained un-damaged throughout training. However, the





(a) (b)

(c) (d)

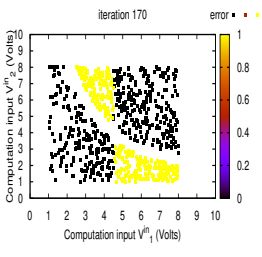
FIGURE 8.4: Microtubules (mT) with H_2O, encapsulated to prevent evaporation (a) during training and (b) after training and mT with LCs (c) before and (d) after training

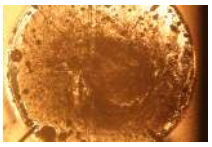
DE algorithm did not converge towards a good solution and neither training nor verification errors were close to the SC problem's optimum. It must be noted that the mixture was prepared without taking into consideration the chemical interactions between LCs and mTs, which might explain the results. Following the discussion reported in [39], a different preparation would be required to ensure a homogeneous dispersion of the mTs within the LC host. The low current levels might remain a factor limiting the use of the mTs in EiM investigations, but this might be remedy to by using samples with higher mT concentrations. The results obtained with layers of mTs extracted from bovine neurons, suggest that the EiM implementation used in experiments, combined with the sample preparation described in this section, resulted in the destruction in the material layer during training which affected the mT samples' ability to solve computational problems.

8.3 Evolving XOR Gates in Memristive Devices

8.3.1 Motivations

The memristor was first postulated in [40] and was suggested to be the fourth fundamental circuit element, linking electric charge and flux linkage in electrical circuits. Figure





8.5 illustrates the four fundamental elements (resistor, capacitor, inductor, memristor), the four fundamental variables (current, voltage, charge, flux) and their relationship. The theoretical formulation and circuit implementation proposed in [40] showed that memristors behave as non-linear resistors with a memory, i.e. hysteresis in the device's I/V relationship. In 2008, such electrical characteristics were measured across a titanium dioxide TiO_2 device [41]. This is generally considered as the first instance of physical realisation of a memristor [42]. The curve was non-linear in its OFF state (increasing voltage) and a sinh-like curve in its ON state (decreasing voltage) [43].

Recent years have seen a large amount of research conducted with the aim of exploring the properties of memristors and their potential in electronics. The combination of TiO_2 and aluminium electrodes is very common, but other materials have also been used to produce devices with memristive properties. It must be noted, however, that the term 'memristor', a contraction of memory and resistor, has become relatively loose and often refers to any device with a non-linear conductance and switching behaviour, irrespective of its mode of operation [42, 44].

One area of investigation within memristor research is neuromorphic circuit design [45]. This is due to the resemblance between the spiking behaviour of the device and that of a synapse. The latter is an element of the neuronal cells present in the brain which allows the transmission of signals between these cells. Devices presenting memristive properties have also been used to produce Boolean logic circuits such as logic gates and half adders [46]. Memristors have not yet been investigated within the EiM framework. However, models of different types of memristors, based on empirical measurements,

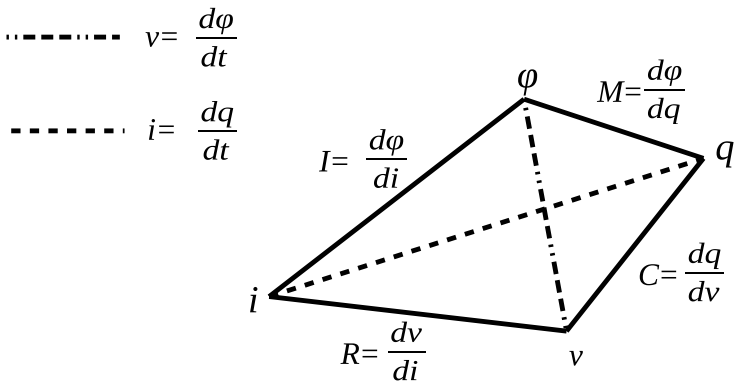
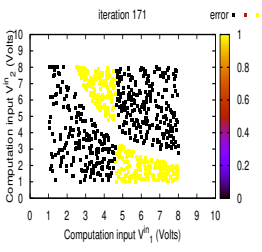
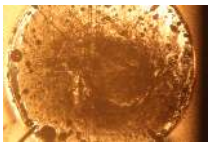


Figure 8.5: Relationship between the four fundamental circuit elements: resistor (R), capacitor (C), inductor (I), memristor (M) and the four fundamental circuit variables: current (i), voltage (v), charge (q) and flux (φ).





have been successfully evolved into robot controllers using EAs [47]. These results, combined with the electrical characteristics of memristors and their potential as an unconventional computing electronic component, make them a material of interest for EiM research.

Sample preparation

Memristors produced at Durham University consisted in a structure of polyfluorene and aluminium (Al/PFO/Al). The PFO was purchased from Merck, Sigma-Aldrich. It is a non-conducting polymer through which wires of Al can form under an applied electric field. The devices were fabricated on microscope slide. Around 100 nm Al was first evaporated onto the slide, through a mask, to form the bottom electrodes. A layer of PFO, 4.5 mg/ml in chloroform, sonicated for 60min, was subsequently spin coated over the slide surface at 1250 rpm for 60 s. Finally a 100 nm thick set of Al electrodes was evaporated above the polymer layer [48]. The top and side views of the devices are illustrated in Figure 8.6, along with the chemical structure of PFO. The bottom and top electrodes were used as negative and positive terminals respectively.

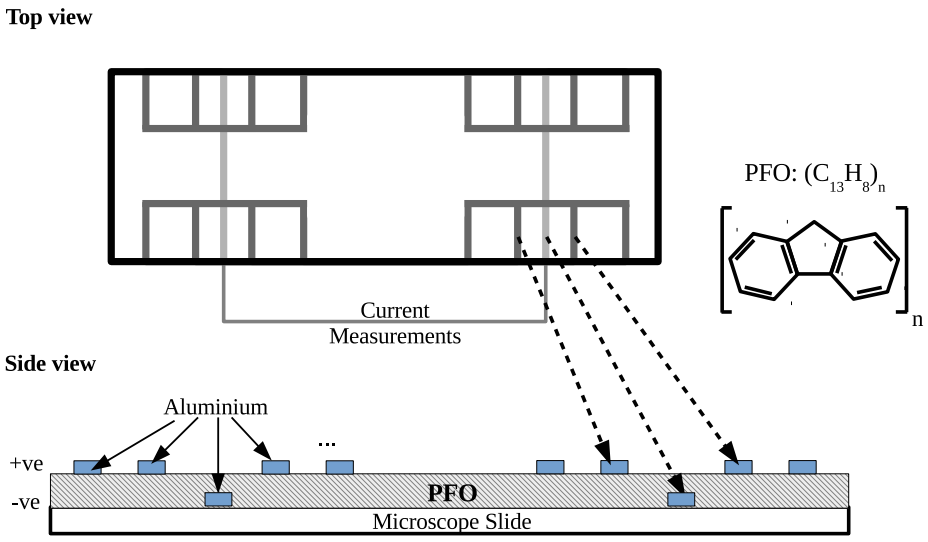
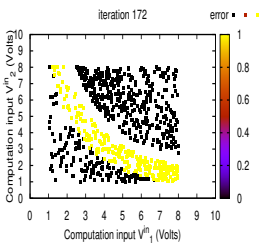
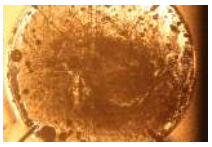


FIGURE 8.6: Top and side illustration of the memristive devices produced at Durham University using a polymer (PFO) sandwiched between to sets of aluminium electrodes of opposite polarity.





Electrical characteristics of Al/PFO/Al memristors

The electrical characteristic of the PFO-Al memristors were tested under vacuum conditions using the source meter. It was noticed that devices left out of the vacuum between tests tended to lose their electrical properties after a short time span. More specifically, they became linear devices with a very high resistance, dropping outputs measured between 0 and 10 V in the nA range. Figure 8.7 presents the I/V relationship measured across four different memristors located on the same microscope slide.

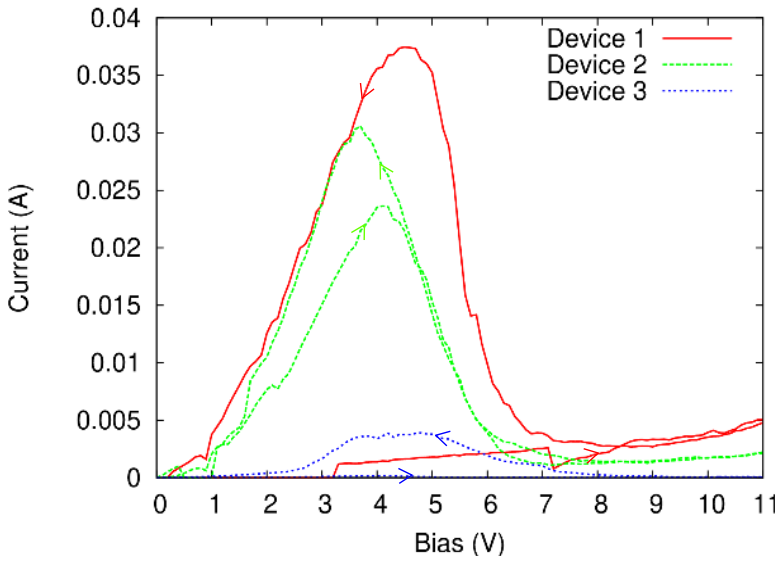
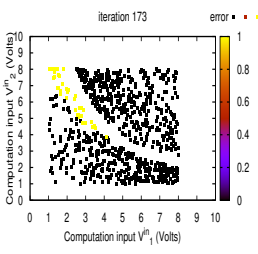


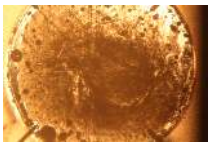
FIGURE 8.7: I/V characteristics of Al/PFO/Al memristor presenting a negative differential resistance.

In this case, each positive aluminium electrode effectively acts as an individual memristor, which means that each slide contains sixteen memristors. It can be observed from the graph that when the devices are in their OFF state, i.e. when voltages are increased up to 11 V, the I/V relationship is non-linear and the memristors' outputs are in the mA range. When the voltages are decreased to 0 V in steps of 0.1 V, the memristors are switched to their ON state and a peak current between $[0.02, 0.04]$ A can be observed around 4 V.

8.3.2 The Boolean Function Problem

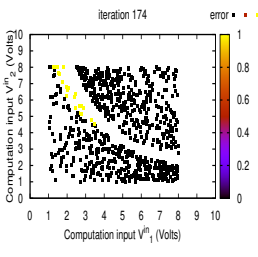
Logic gates such as the **AND**, **OR** and **XOR** gates are the basic components used in boolean logic. Any Turing computable function can be modelled using only these three gates [49]. It is therefore common when exploring new computing approaches to test whether the basic logic gates can be realised. The **XOR** problem has the advantage of

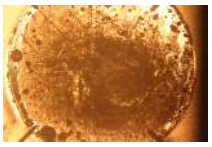




having been solved with different EiM implementations [13, 14, 50, 51]. This suggests that it is possible to solve this problem using EiM, making it a good test of the potential of a new material or implementation. The widespread use of this problem in EiM also provides means for comparison within the field, rather than against methods implemented *in silico*. In addition, results reported in [52] demonstrate the possibility to produce an **XOR** gate using a single TiO_2 memristor to which a series of DC voltages have been applied sequentially. The **XOR** gate was realised using implication logic, a structured mathematical framework not unlike Turing Logic. This differs from an EiM approach where the series of DC voltages would be controlled by an evolutionary algorithm. It is observed in [52] that the time taken to perform the Boolean function, i.e. for the memristor to act as an **XOR** gate under the application of two inputs, is large as compared to *in silico* electronic components. However, it is argued that this time limitation is balanced by the fact that a single memristor was sufficient to produce the output of an **XOR** gate, where multiple transistors would be needed.

In the work presented here the problem of evolving a memristor into a device capable of acting as an **XOR** gate is preferred to that of evolving this material into a device capable of classifying data, the type of problem investigated previously. This choice was mainly motivated by the fact that the ability of memristors to perform this type of function is not a trivial problem, that it has already been demonstrated *in materio* using a different method [52], and so has the ability of the EiM process to transform materials into **XOR**-gates. In addition, the fact that in EiM the inputs are translated into voltages means that evolving a material to behave as an **XOR** gate is not a trivial task. This is especially the case because the lowest and highest energy input combination (1,1) need to be assigned the same state (low, i.e. 0). However, one characteristic of memristive devices is an I/V behaviour exhibiting a negative differencial resistance effect. This means that past a given input voltage level, the device will present decreasing current outputs at increasing voltage inputs. This behaviour is thought to make memristors particularly well suited to perform an **XOR** function. Here, the aim is to transform the Al/PFO/Al memristors into **XOR** gates using the learning power of evolutionary algorithms, combined with the learning power of the memristor itself. This might pose a problem as the material's memory is based on the sequential application of electrical input. In other words, the algorithm implicitly assumes that the substrate is memoryless - which is not





true in this case, and only partially so in the dynamic SWCNT-based composites.Poor solutions tested by the algorithm may create 'confusion' in the memristor. The output currents measured across the memristors as they are being evolved have therefore been recorded in order to measure the variations in the materials electrical characteristics.

XOR problem description

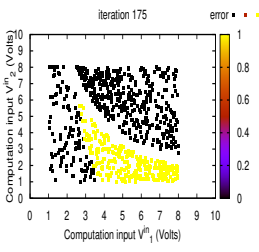
In the simple definition of a computation as a top level input/output process, the input of the **XOR** gate is a vector of $n_1 = 2$ characteristic features, containing the different combinations of highs and lows (1s and 0s, respectively). The output is the resulting high or low state which is assigned to the combination sent as input according to Table 8.3. The array of computation voltages is populated by the two voltages representing input 1 and input 2, $\mathbf{V}^{\mathcal{C}} = [V_1^{\mathcal{C}}, V_2^{\mathcal{C}}]$. The output of evolved memristor-based **XOR**-gate is a measure of the material's state, **Y**(**M**), under the influence of the computation inputs and the set of configuration voltages selected by the algorithm.

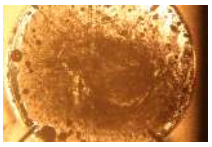
TABLE 8.3: **XOR** gate truth table for a two input vector

Input 1 (V_1^C)	Input 2 (V_2^C)	Output (**Y**(**M**))
0	0	0
0	1	1
1	0	1
1	1	0

The supervised learning approach described in Chapter3, Section 3.2 is followed to solve the optimisation problem. In this case, the **XOR** computational problem is effectively treated as a binary classification problem. A discriminative method is used to produce a *discriminant function* $f\left(\mathbf{V}^{\mathcal{C}}\right)$ mapping directly $\mathbf{V}^{\mathcal{C}}$ to one of the states (i.e. classes) $\mathcal{A}_i$. Similarly to the classification problem, when the EiM approach is used, the discriminative function is replaced by a material. The material's state is modified until it is able to discriminate between a high or a low when an input is applied following the combination in Table 8.3, i.e. until it responds to these inputs as an **XOR** logic gate.

The problem is binary, i.e. it has two dimensions, $\mathcal{D} = \{1, 2\}$ and $\mathcal{A}$ consists of two subspaces $\mathcal{A}_1$ and $\mathcal{A}_2$, corresponding to the low (0) and high (1) states, with $\mathcal{A}_1 \cup \mathcal{A}_2 = \mathcal{A}$. In this sense, the two states are equivalent to two separable classes in the classification problem, and $\mathcal{A}_1 \cap \mathcal{A}_2 = \emptyset$. The resulting binary classifier is given computation inputs





$\mathbf{V}^{\mathcal{C}} \in \mathcal{A}$ and assigns them to either state 0 or state 1. In this sense the computation performed is

$$\mathcal{C}\left(\mathbf{V}^{\mathcal{C}}\right) = \begin{cases} 1 & \text{if} \quad \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_1 \\ 2 & \text{if} \quad \mathbf{V}^{\mathcal{C}} \in \mathcal{A}_2, \end{cases} \tag{8.1}$$

where $\mathcal{C}(\mathbf{V}^{\mathcal{C}})$ is the class, or state, 0 or 1 produced when an instance is sent to the material under the form of two input voltages, $V_1^{\mathcal{C}}, V_2^{\mathcal{C}}$.

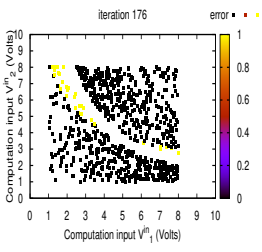
8.3.3 Experimental Implementation

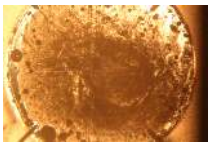
The memristors experiments were implemented using the supervised learning approach, where input voltages, corresponding to the attributes of instances from the training or verification datasets, and a number of configuration voltages controlled by an algorithm are applied to the device. The number of computation voltages was limited to two, i.e a two input **XOR** gate, whilst the number of configuration voltages was limited to 4, due to the available electrodes.

Before each training, memristors were subjected to a voltage sweep, in order to monitor any potential degradation in its I/V characteristics across experiments. Before each new iteration, the material was left with no applied voltages for 40 seconds. The training dataset contained $K_t = 50$ instances, each defined by two 0's, two 1's, or a combination of 0 and 1. The exact values for the 0's and 1's were 0V and 2V in order to allow for potential noise corrupting the inputs. The verification dataset contained the same number of instances: $K_v = 50$, and was applied five minutes after the end of training, which means that the memory acquired during training had dissipated towards the end of the process. The optimal solution was then applied to the material, followed by the verification instances.

Problem formulation

In this case two thresholds, $R_1, R_2 \in \mathbf{R}$ were used to separate the data, and one output was measured across the device, I_1. The resulting interpretation scheme was of the form:





$$S_{\mathcal{C}}^{XOR}(\mathbf{Y}) = \begin{cases} 0 & \text{if} \quad I_1 \geq R_1 \\ 1 & \text{if} \quad R_1 \leq I_1 < R_2 \\ 0 & \text{if} \quad I_1 \geq I_2 \end{cases} \tag{8.2}$$

The a basic outline of the implementation and the scheme employed is presented in Figure 8.8

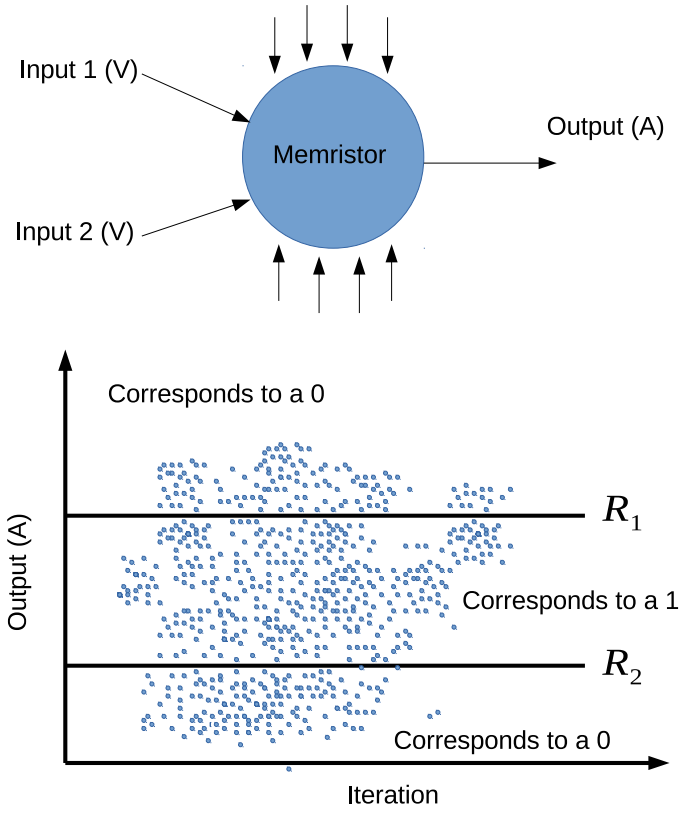
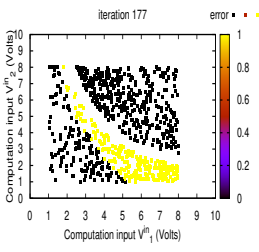
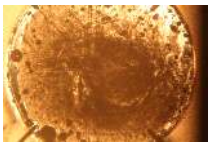


FIGURE 8.8: Implementation of the interpretation scheme used to evolve Al/PFO/Al memristors into **XOR** gates

Algorithms parameters

Both DE and PSO were used here and their parameters are the same as those previously used in experiments (see Chapter 4, Section4.5). The search parameters selected for these investigations are reported in Table 8.4. The minimum and maximum values for the two thresholds, R_1 and R_2 are based on the data collected during the electrical characterisation of the devices, where the maximum current obtained, during the spike, is 12mA. Similarly, the maximum configuration voltage is increased to 6 V and the input electrodes are kept the same throughout training.





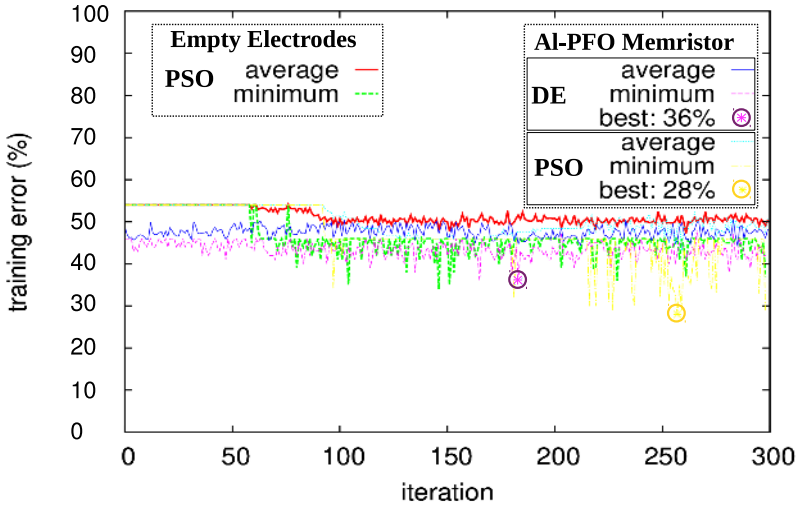


FIGURE 8.9: Comparison of average and minimum training errors per iteration, along with the overall best training error for the **XOR** problem between control sample (empty electrodes), memristor trained with the DE algorithm and memristor trained with the PSO algorithm

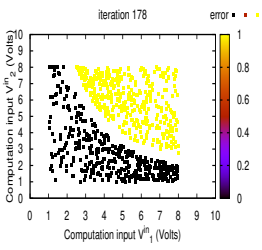
8.3.4 Results

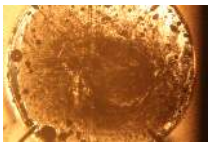
First of all, it was observed that the material's electrical characteristics degraded quickly when it was trained in the laboratory at a constant temperature of $20°C$. In this case, no conclusive results were obtained from the evolution of the devices. The EiM experiments were therefore repeated with the memristor placed with a vacuum chamber at constant pressure, such that the device's electrical characteristics would not be affected from its exposure to the environment throughout training.

The objective function average and minimum per iteration remained within 10% of the worst error, i.e the 50% error obtained with an unconfigured material. This can be observed in Figure 8.9 where the convergence of the objective function, in terms of average and minimum error per iteration, are reported for DE and PSO-trained memristors, and compared to the case where an empty electrode array has been trained using PSO for the same problem.

TABLE 8.4: Search Parameters.

	Parameter	Value
Search	iteration size (Λ)	300
	population size (N)	5
	number of configuration voltages (n_2)	4
	$[V_{min}, V_{max}]$ (Volts)	[0, 6]
	$[R_{1,min}, R_{1,max}]$	[8, 12]
	$[R_{2,min}, R_{2,max}]$	[0.05, 4]





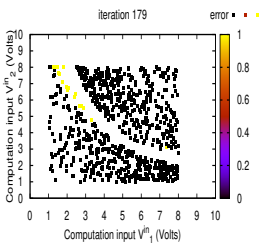
Results for the later tests can be found in Table 8.5. In addition, the electrical characteristics were found to deteriorate under the repeated application of an electric field. Contrary to results reported in [14] and [53] with SWCNT/PBMA composites, it was not possible to evolve a memristor into a state where it could behave as an **XOR** gate following the EiM implementation used here. The minimum training error, averaged of the five experiments, was 35.677% for DE and 28% for PSO, which means that in both cases, more than 20% of the 50 training instances were assigned the incorrect output value. Errors obtained during verification are around 50%, which means that 50% of the instances contained in the verification dataset have been process in a way that does not follow the way an **XOR** should function. As observed in Figure 8.9 with the training error, the verification errors obtained with the memristors are the same as those obtained with the empty electrode array.

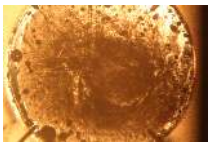
TABLE 8.5: Comparing training and verification performance between memristors XOR gate and SWCNT/PBMA XOR-gates, both evolved through classical EiM

Algorithm	Material	$\Phi^*_e(\%)$	$\Phi^*_{e,v}(\%)$	$\overline{\Phi}_{e,v}$	$\sigma(\%)$
DE	memristor	35.677	55.000	55.000	0.000
PSO	memristor	28.000	55.000	55.000	0.000
GA [53]	SWCNT/PBMA	0.000	0.000	0.000	N/A
PSO [14]	SWCNT/PBMA	0.000	0.000	0.000	N/A

It must be noted that despite having placed the memristors in the vacuum chamber, results did not improve. In addition, the memristors tended to lose their characteristic electrical behaviour, as illustrated in Figure 8.10. The current level has gone down after training, and the voltage spike around 4V can no longer be observed.

Figure 8.11 compares the current/voltage (I/V) characteristics of a memristor as it is measured by the evolvable motherboard (EM) and the Keithley 2400 digital source meter. The latter has been used to characterise the electrical behaviour of all materials investigated here (see Chapter 2, Section 2.3.1). In both cases, the material is under vacuum. It can be observed that the EM has a lower sensitivity to the memristor and the current output measured across the device is noisy compared to the measurements collected with the Keithley source meter. This is an issue in the sense that the outputs, transformed by the interpretation scheme, will be affected by the noise, and might therefore not always reflect the actual state of the material.





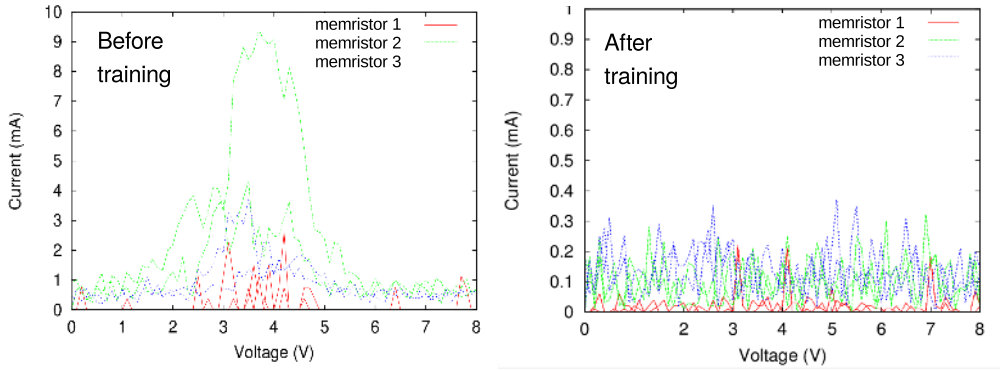


FIGURE 8.10: Comparison of memristor current / voltage characteristics before and after training, measured across the evolvable motherboard.

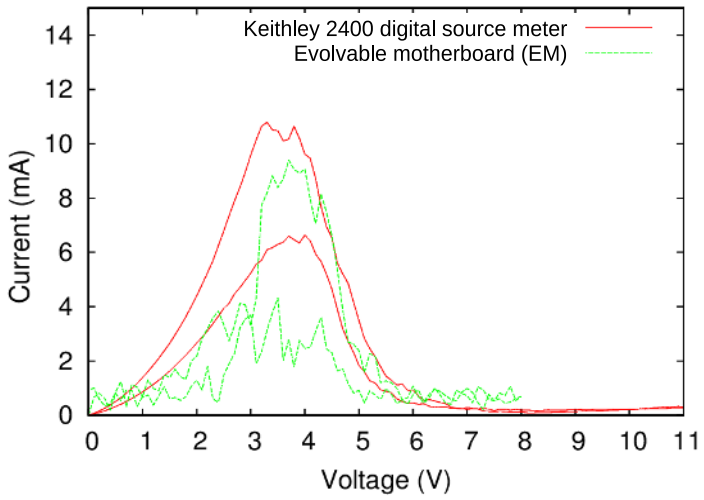


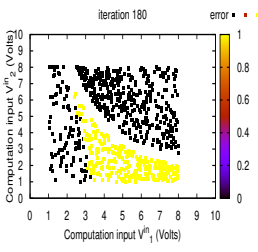
FIGURE 8.11: Comparison of memristor current / voltage characteristics collected by the evolvable motherboard before (top gaph) and after (bottom graph) training.

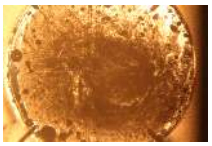
8.4 Summary of Results and Conclusions

This chapter has presented two new avenues for research within the field of evolution in materio (EiM), based on two materials, microtubules and memristors, which had never previously been investigated within this context:

Microtubules were chosen as a result of a short-term collaboration with Prof. Horacio Cantello. They present an alternative to silicon with a biological origin, setting them apart from the other materials used here. Yet, they are simpler structures than the micro-organisms generally used in biological computing, and importantly their electrical characteristics fit the criteria required for EiM allowing them to be treated similarly to the non-biological samples.

Memristors have been studied extensively since memristive behaviour [40] was





found to be an inherent property of materials at nanoscale [41]. More recently, models and physical devices have been increasingly used in conjunction with artificial neural networks (ANNs), due to the similarity of their electrical properties to that of switching synapses. Memristors fit the requirements of EiM, are relatively new and research has demonstrated their capacity to perform computational tasks [52] and to be integrated with current technology [54].

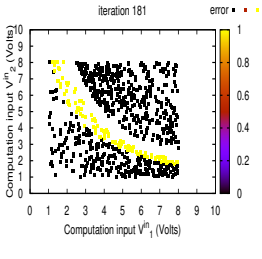
The biological substrate, mTs, were the first subject of investigation in this chapter. Differential evolution (DE) and particle swarm optimisation (PSO) were used to find solutions to a simple binary classification problem and to a benchmark medical classification problem in this biological substrate. Whilst it was not possible to reach optimal results using the microtubule thin-film, two behaviours were observed: 1) the material tended to dry throughout training, resulting in samples with next to no output current and 2) encapsulating the material to prevent the evaporation destroyed the film. These two observations need to be taken into consideration by anyone interested in using microtubules as a material for evolution in materio.

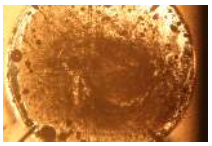
The second material, memristor, was trained using the two algorithms to behave as an Exclusive OR (**XOR**) gate, i.e. perform a relatively complex Boolean function. The material's properties deteriorated in the clean-room environment. Storing the memristive devices in a vacuum chamber allowed the memristor's outputs to remain high enough to be measurable using the evolvable motherboard, a necessary requirement for the material to be evolved using the EiM framework. However, the training resulted in devices which were not capable of behaving as **XOR**-gate. The important variations induced by the algorithms during training destroyed the memristive properties after some time. It would be interesting to investigate the use of memristors within the context of EiM, but rather in an array where algorithms modify the connections between memristors rather than the level of the input sent to the memristors themselves.

Bibliography

[1] J. F. Miller and K. Downing, "Evolution in materio: Looking beyond the silicon box," *Evolvable Hardware, 2002. Proceedings. NASA/DoD Conference on*, pp. 167–176, 2002.

[2] S. Bose, C. P. Lawrence, Z. Liu, K. Makarenko, R. M. van Damme, H. J. Broersma, and W. G. van der Wiel, "Evolution of a designless nanoparticle network into reconfigurable boolean logic," *Nature nanotechnology*, vol. 10, no. 12, p. 1048, 2015.





[3] H. J. Broersma, F. Gomez, J. Miller, M. Petty, and G. Tufte, "Nascence project: Nanoscale engineering for novel computation using evolution," *International journal of unconventional computing*, vol. 8, no. 4, pp. 313–317, 2012.

[4] NASCENCE project (ICT 317662). http://nascence.no/, 2015. NAnoSCale Engineering for Novel Computation using Evolution.

[5] H. Broersma, *Evolution in Nanomaterio: The NASCENCE Project*, pp. 87–111. Cham: Springer International Publishing, 2018.

[6] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving machine learning classification problems using materials," pp. 721–730, Springer International Publishing, 2014.

[7] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebo, M. K. Massey, and M. C. Petty, "Evolution-in-materio: A frequency classifier using materials," in *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 46–53, IEEE, 2014.

[8] M. Mohid, J. F. Miller, S. L. Harding, G. Tufte, O. R. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving function optimization problems using materials," in *Computational Intelligence (UKCI), 2014 14th UK Workshop on*, pp. 1–8, IEEE, 2014.

[9] M. Mohid, J. Miller, S. Harding, G. Tufte, O. Lykkebø, M. K. Massey, and M. C. Petty, "Evolution-in-materio: Solving bin packing problems using materials," *Evolvable Systems (ICES), 2014 IEEE International Conference on*, pp. 38–45, 2014.

[10] M. Mohid and J. Miller, "Solving even parity problems using carbon nanotubes," in *Computational Intelligence (UKCI), 15th UK Workshop on. IEEE Press*, 2015.

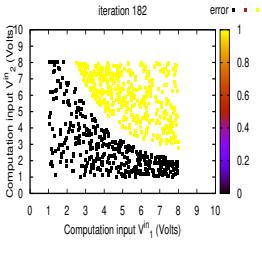
[11] M. Mohid and J. F. Miller, "Evolving solutions to computational problems using carbon nanotubes.," *International journal of unconventional computing*, vol. 11, 2015.

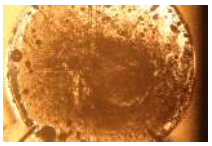
[12] S. S. Farstad, S. Nichele, and G. Tufte, "Towards standalone in-materio devices: Stable logic gates and elementary cellular automata in carbon nanotubes material," in *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 5246–5254, IEEE, 2016.

[13] F. Qaiser, A. Kotsialos, M. K. Massey, D. A. Zeze, C. Pearson, and M. C. Petty, "Manipulating the conductance of single-walled carbon nanotubes based thin films for evolving threshold logic circuits using particle swarm optimisation," in *IEEE Congress on Evolutionary Computation (CEC)*, pp. 5255–5261, IEEE, 2016.

[14] A. Kotsialos, M. K. Massey, F. Qaiser, D. A. Zeze, C. Pearson, and M. C. Petty, "Logic gate and circuit training on randomly dispersed carbon nanotubes.," *International Journal of Unconventional Computing*, vol. 10, no. 5-6, pp. 473–497, 2014.

[15] M. Massey, A. Kotsialos, F. Qaiser, D. Zeze, C. Pearson, D. Volpati, L. Bowen, and M. Petty, "Computing with carbon nanotubes: Optimization of threshold logic gates using disordered nanotube/polymer composites," *Journal of Applied Physics*, vol. 117, no. 13, p. 134903, 2015.





[16] D. Matthew, J. F. Miller, S. Stepney, and M. A. Trefzer, "Evolving carbon nanotube reservoir computers," *Unconventional Computation and Natural Computation: 15th International Conference, UCNC 2016*, pp. 49–61, 2016.

[17] M. Dale, *Reservoir Computing In-Materio (PhD Thesis)*. University of York, 2018.

[18] S. L. Harding, J. Neil, K. Zauner, and J. Miller, "A Framework for the Automatic Identification and Extraction of Computation from Materials," *Computer*, 2008.

[19] D. Laketić, G. Tufte, S. Nichele, and O. R. Lykkebø, "Bringing colours to the black box–a novel approach to explaining materials for evolution-in-materio," in *Proceedings of 7th International Conference on Future Computational Technologies and Applications. XPS Press*, 2015.

[20] S. Harding, *Evolution in Materio (PhD Thesis)*. The University of York, 2006.

[21] K. Greff, R. van Damme, J. Koutnik, H. Broersma, J. Mikhal, C. Lawrence, W. van der Wiel, and J. Schmidhuber, "Unconventional computing using evolution-in-nanomaterio: neural networks meet nanoparticle networks," in *Eighth International Conference on Future Computational Technologies and Applications, FUTURE COMPUTING 2016*, pp. 15–20, 2016.

[22] M. Amos, I. M. Axmann, N. Blüthgen, F. de la Cruz, A. Jaramillo, A. Rodriguez-Paton, and F. Simmel, "Bacterial computing with engineered populations," *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, vol. 373, 2015.

[23] J. Jones, J. Whiting, and A. Adamatzky, "Quantitative transformation for implementation of adder circuits in physical systems," *Biosystems*, vol. 134, pp. 16–23, 2015.

[24] M. W. Rochlin, M. E. Dailey, and P. C. Bridgman, "Polymerizing microtubules activate site-directed f-actin assembly in nerve growth cones," *Molecular Biology of the Cell*, no. 10, pp. 2309–2327, 1999.

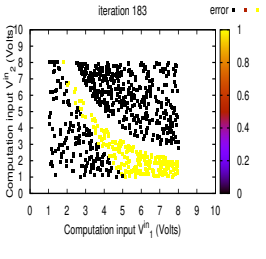
[25] P. W. Baas, A. N. Rao, A. J. Matamoros, and L. Leo, "Stability properties of neuronal microtubules," *Cytoskeleton*, vol. 73, no. 9, pp. 442–460, 2016.

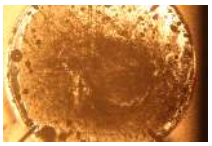
[26] B. Lacroix, G. Letort, L. Pitayu, J. Sallé, M. Stefanutti, G. Maton, A.-M. Ladouceur, J. C. Canman, P. S. Maddox, A. S. Maddox, *et al.*, "Microtubule dynamics scale with cell size to set spindle length and assembly timing," *Developmental cell*, vol. 45, no. 4, pp. 496–511, 2018.

[27] S. Hameroff and R. Penrose, "Orchestrated reduction of quantum coherence in brain microtubules: A model for consciousness," *Mathematics and computers in simulation*, vol. 40, no. 3-4, pp. 453–480, 1996.

[28] S. Hameroff, "Quantum computation in brain microtubules? the penrose-hameroff'orch or'model of consciousness," *Philosophical Transactions-Royal Society of London Series A Mathematical Physical and Engineering Sciences*, pp. 1869–1895, 1998.

[29] S. Hagan, S. R. Hameroff, and J. A. Tuszyński, "Quantum computation in brain microtubules: Decoherence and biological feasibility," *Physical Review E*, vol. 65, no. 6, p. 061901, 2002.





[30] L. K. McKemmish, J. R. Reimers, R. H. McKenzie, A. E. Mark, and N. S. Hush, "Penrose-hameroff orchestrated objective-reduction proposal for human consciousness is not biologically feasible," *Phys. Rev. E*, vol. 80, p. 021912, 2009.

[31] S. Hameroff and R. Penrose, "Consciousness in the universe: A review of the 'orch or'theory," *Physics of life reviews*, vol. 11, no. 1, pp. 39–78, 2014.

[32] J. R. Reimers, L. K. McKemmish, R. H. McKenzie, A. E. Mark, and N. S. Hush, "The revised penrose–hameroff orchestrated objective-reduction proposal for human consciousness is not scientifically justified: Comment on "consciousness in the universe: a review of the 'orch or'theory" by hameroff and penrose," *Physics of life reviews*, vol. 11, no. 1, pp. 101–3, 2014.

[33] A. Priel, A. J. Ramos, J. A. Tuszynski, and H. F. Cantiello, "A biopolymer transistor: electrical amplification by microtubules," *Biophysical journal*, vol. 90, no. 12, pp. 4639–4643, 2006.

[34] M. del Rocío Cantero, P. L. Perez, M. Smoler, C. V. Etchegoyen, and H. F. Cantiello, "Electrical oscillations in two-dimensional microtubular structures," *Scientific Reports*, vol. 6, p. 27143, 2016.

[35] M. del Rocío Cantero, C. V. Etchegoyen, P. L. Perez, N. Scarinci, and H. F. Cantiello, "Bundles of brain microtubules generate electrical oscillations," *Scientific reports*, vol. 8, no. 1, p. 11899, 2018.

[36] A. Chiolerio, T. C. Draper, R. Mayne, and A. Adamatzky, "On resistance switching and oscillations in tubulin microtubule droplets," *Journal of colloid and interface science*, vol. 560, pp. 589–595, 2020.

[37] M. W. Davidson and T. F. S. University. https://micro.magnet.fsu.edu/cells/microtubules/microtubules.html, 2015.

[38] M. Lichman, "UCI machine learning repository," 2013.

[39] A. Lackner, K. Lim, J. Margerum, and E. Sherman, "Microtubule particle dispersion in liquid crystal hosts," *Liquid Crystals*, vol. 14, no. 2, pp. 351–359, 1993.

[40] L. Chua, "Memristor-the missing circuit element," *IEEE Transactions on circuit theory*, vol. 18, no. 5, pp. 507–519, 1971.

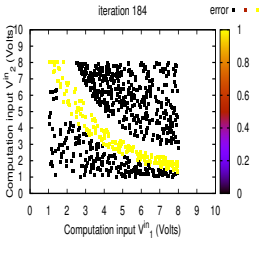
[41] D. B. Strukov, G. S. Snider, D. R. Stewart, and R. S. Williams, "The missing memristor found," *nature*, vol. 453, no. 7191, p. 80, 2008.

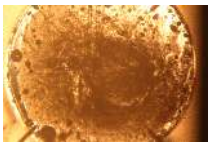
[42] E. Gale, "Tio 2 -based memristors and reram: materials, mechanisms and models (a review)," *Semiconductor Science and Technology*, vol. 29, no. 10, p. 104004, 2014.

[43] J. J. Yang, M. D. Pickett, X. Li, D. A. Ohlberg, D. R. Stewart, and R. S. Williams, "Memristive switching mechanism for metal/oxide/metal nanodevices," *Nature Nanotechnology*, vol. 3, no. 7, p. 429, 2008.

[44] M. C. Petty, *Organic and Molecular Electronics: From Principles to Practice*. New York, NY, USA: Wiley, 2018.

[45] L. Xu, C. Li, and L. Chen, "Analog memristor based neuromorphic crossbar circuit for image recognition," in *Intelligent Control and Information Processing (ICICIP), 2015 Sixth International Conference on*, pp. 155–160, 2015.





[46] E. Gale, "Single memristor logic gates: From not to a full adder," *arXiv preprint arXiv:1510.05705*, 2015.

[47] G. Howard, L. Bull, B. de Lacy Costello, E. Gale, and A. Adamatzky, "Evolving spiking networks with variable resistive memories," *Evol. Comput.*, vol. 22, pp. 79–103, Mar. 2014.

[48] T.-W. Kim, K. Lee, S.-H. Oh, G. Wang, D.-Y. Kim, G.-Y. Jung, and T. Lee, "A direct metal transfer method for cross-bar type polymer non-volatile memory applications," *Nanotechnology*, vol. 19, no. 40, p. 405201, 2008.

[49] S. G. Akl, "Nonuniversality in computation: fifteen misconceptions rectified," in *Advances in Unconventional Computing*, pp. 1–30, Springer, 2017.

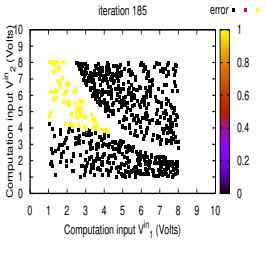
[50] S. L. Harding and J. F. Miller, "Evolution in materio: Evolving logic gates in liquid crystal," *Proc. Eur. Conf. Artif. Life (ECAL 2005), Workshop on Unconventional Computing: From cellular automata to wetware*, pp. 133–149, 2005.

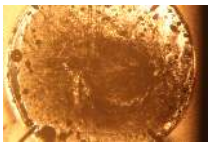
[51] M. Mohid, *Evolution-In-Materio: Solving Computational Problems Using Materials (PhD Thesis)*. University of York, 2015.

[52] E. Gale, B. de Lacy Costello, and A. Adamatzky, "Boolean logic gates from a single memristor via low-level sequential logic," *Unconventional Computation and Natural Computation: 12th International Conference, UCNC 2013*, pp. 79–89, 2013.

[53] O. R. Lykkebø, S. Harding, G. Tufte, and J. F. Miller, "Mecobo: A hardware and software platform for in materio evolution," in *International Conference on Unconventional Computation and Natural Computation*, pp. 267–279, Springer, 2014.

[54] A. Serb, A. Khiat, and T. Prodromakis, "Seamlessly fused digital-analogue reconfigurable computing using memristors," *Nature Communications*, vol. 9, no. 1, p. 2170, 2018.





Chapter 9
Conclusions

9.1 Research Hypothesis - Recap

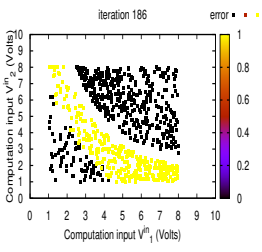
The research hypothesis motivating the investigations presented here is that, within the context of evolution in materio (EiM), a dynamic state in single-walled-carbon-nanotube (SWCNT)-based composites can provide an extra layer of complexity as compared to their solid SWCNT-based counterpart. This complexity can induce unforeseen advantages to the evolved devices.

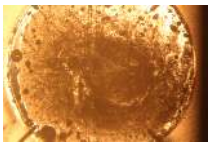
9.2 Chapters and Contributions Summary

Chapter 4

The ability of the EiM framework to transform SWCNT/LC samples into data classifiers was first demonstrated using two different evolutionary algorithms (EA): differential evolution (DE) and particle swarm optimisation (PSO). Both algorithms were able to evolve at least one sample into a device able to classify data with 100% accuracy.

It was observed that the EA's ability to transform the composites successfully into data classifiers depended on the SWCNT concentration. Further investigations suggested that optimum training, in terms of speed and result accuracy, were obtained with composites of concentration between 0.05 wt % and 0.1 wt % SWCNT/LC. This optimum range of SWCNT/LC concentrations is lower than the optimum SWCNT concentration reported for solid SWCNT/polymer composites.





The changes in the bulk material morphology observed during the SWCNT/LC samples' training suggested that the EAs were not only exploring and exploiting an existing SWCNT network as they do in solid SWCNT-based composites, but also transforming this network throughout training, thereby expanding the algorithm's search space.

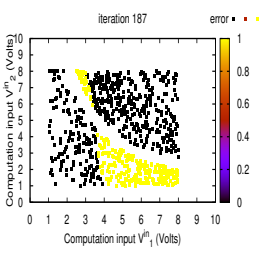
Differences in training efficiency, speed, result reproducibility and solution quality were observed between samples, depending on the algorithm with which they were evolved. The dependence of training efficiency on the EAs' search behaviour was therefore investigated. The DE algorithm's search focused mainly on exploitation and resulted in devices classifying data with almost perfect accuracy, but with a lower reproducibility than PSO. The latter evolved devices which were not as accurate, but these results were highly reproducible. It was suggested that the difference in results accuracy and reproducibility was the consequence of the difference in the configuration voltage trajectories applied to the material throughout training.

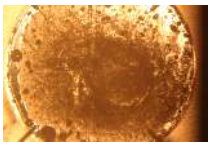
The DE algorithm produced noisy voltages with sharp changes between iterations, likely to destroy a SWCNT-structure favouring a good classifying state. On the other hand, the PSO algorithm produced sinusoidal-like voltage trajectories with minimal change from iteration to iteration. This behaviour suggested that PSO's search was less likely to destroy potentially good solutions. Another hypothesis was that the difference in results between the two algorithms was based on the level of the electric field applied throughout training, which was algorithm-dependent. In either case, PSO's exploration-focused search seldom resulted in optimal solutions in terms of classifier accuracy. It was therefore decided to modify implementation constraints to reduce the amplitude of change in the DE-controlled configuration voltages and increase the number of individuals in the DE population and use mainly this algorithm in subsequent experiments.

The choice of parameters used in the formulation of the EiM training problem also affected the experimental results. However, it was not possible to link changes in parameters with changes in material behaviour during training.

Chapter 5

Interestingly, it was observed that using the DE algorithm, the SWCNT/LC samples' evolved state allowed data to be classified without the need for an applied electric field,





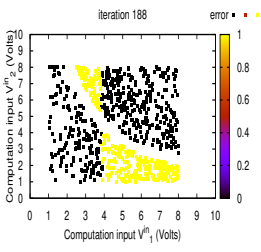
other than that provided by the data itself. In other words, the training had evolved a classifying SWCNT/LC device rather than optimised a set of voltages making a SWCNT/LC sample behave as a classifier. In addition, whilst solution stability was an issue in liquid crystal display (LCD) experiments, as discussed in [1], the classifying state of the SWCNT/LC evolved samples was stable for a number of hours with little deterioration in the classification error.

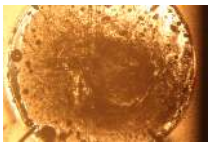
Results obtained with the SWCNT/LC composite suggested that the stability of the state resulted from the SWCNT structures produced by the algorithm's search process. It was observed that this stability tended to depend on the complexity of the classification problem which the samples were evolved to solve. More training, i.e. more changes in the SWCNT structures via the repeated application of EA-controlled voltages, was required to bring the composites into a state able to solve the more complex problems. In return, the classifying state was more stable in those composites, as compared to those evolved to solve the less complex problems. Similarly, the ability to classify data without configuration voltages was only possible in devices trained to solve the more complex problems. The structures are reminiscent of the iron thread grown by Gordon Pask in ferrous sulfate ($FeSO_4$) to perform tone discrimination [2], albeit in the case of the SWCNT/LC composite, the structures are the results of nanotube *bundling* and *rearrangement*, rather than the *growth* of intertwined metallic wires.

It was noted that the classification error obtained by evolving SWCNT/LC samples using evolution in materio (EiM) deteriorated with time, as the material relaxed to its original state, and despite the stability suggested by the presence of the evolved SWCNT-structures. In addition to time, shaking the sample, dropping it, etc, resulted in partial or complete loss of the solution, i.e. a return of the classification error to that obtained in the samples pre-training.

Chapter 6

The conclusions of the SWCNT/LC experiments lead to a series of investigations into new SWCNT composites capable of being evolved whilst liquid, and subsequently solidified in order to encapsulate the evolved SWCNT structure. Results obtained with SWCNT/epoxy devices, a material used for the first time in EiM investigations were promising. It was observed that one of the liquid composites presented the same rate





of change in morphology as SWCNT/LC composites evolved for the same classification problem. In the case of that composite, it was also possible to minimise the classification error during training, and produce devices capable of classifying unseen data with relatively high accuracy. The solidification, or curing, process resulted in low deterioration of the classifiers' accuracy, suggesting that the method is viable. If it was compared to the SWCNT/polymer devices, these hybrid composites would be able to compute data, once evolved and solidified, without the need for an optimum set of configuration voltages. The end device would therefore be more energy efficient. However, the training and verification errors obtained with the SWCNT/epoxy samples were not as good as those obtained with SWCNT/LC samples. In addition, at this stage, the reproducibility of results was low.

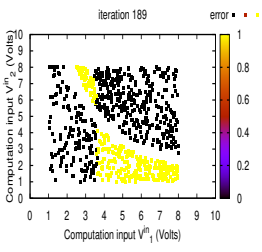
Chapter 7

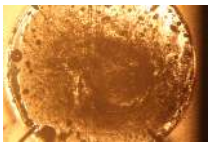
Following the results presented and discussed in Chapters 4-6, the next logical step in the investigations was to verify whether liquid SWCNT-based samples were capable of being evolved to solve real-life problems. Datasets defining three different real-life problems were retrieved from the UCI repository: Iris and mammographic mass. Since the results obtained with SWCNT/epoxy composites were not yet as accurate as those obtained with SWCNT/LC samples, it was decided to use the latter in the real-life dataset investigations.

It was observed that DE-trained SWCNT/LC classifiers were able to classify data from the Iris dataset an accuracy comparable with that obtained with the SWCNT/PBMA classifiers produced with following the classical EiM process, but implemented in the Mecobo board and using a different algorithm and problem formulation.

It was not possible to obtain a better accuracy that the reservoir computing in materio (RCiM) implementation or algorithms run on conventional computers (*in silico*). Results obtained for the mammographic dataset were compared with *in silico* training of different types of neural networks (NN) using the DE algorithm, as well as the diagnostics formulated by human radiologists.

The evolved SWCNT/LC classifiers were able to classify mammographic masses with an error % slightly larger than non-fellowship trained radiologists. However, the classification accuracy was worst that for fellowship-trained radiologists and other NN





implementations. Overall, it was possible to produce devices capable of classifying complex real-life data with a relative accuracy, but the combination of EiM/EA/material was not comparable with state-of-the art algorithms.

Chapter 8

Two other new materials for EiM were investigated. The first was based on microtubules extracted from bovine neurons were used to solve first the simplest artificial data classification problem, then one of the complex real-life problems. The idea of investigating this new type of material, of biological origin was proposed by Prof. Horacio Cantiello. The choice of material was justified by the fact that they present a similar aspect ratio to SWCNTs, viable electrical conductivity and have been found to play an essential part in information transmission within the cells.

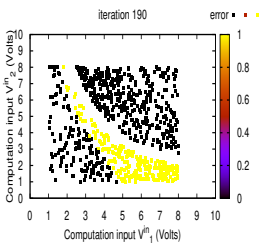
Results obtained with this material were very poor, irrespective of the complexity of the classification problem, the EA used or the microtubule concentration. Multiple attempts at improving the general training efficiency were unsuccessful. EiM training of the rehydrated microtubule films drop-cast on a micro-electrode array tended to destroy the material irreversibly.

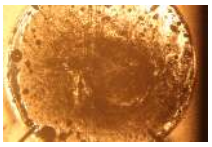
The other material explored was the memristor, which has recently seen a rise in popularity amongst computer scientists and engineers. In this case, the DE algorithm was used to evolve an **XOR** gate out of a memristor using the same implementation used for the SWCNT-based composite and the microtubule. However, this implementation was not well suited to the memristor. It was not possible to evolve it into **XOR** gate, and in addition, a device lost its memristive properties after one experiment.

9.3 Further Work

9.3.1 EA Library and their Associated Impact on Dynamic Composites

It was observed in Chapter 3 that the EA search behaviour had an important impact on the reproducibility and solution quality of evolved SWCNT/LC classifiers. Results suggested that this impact was caused by the way the EA search affected the SWCNT/LC composite's morphology throughout training. Two EAs were compared in this work. It is possible that algorithms with different search strategies would produce devices with new





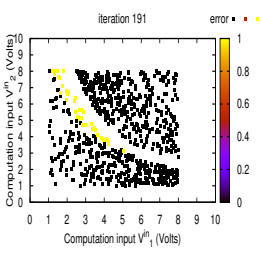
and interesting characteristics, or optimise existing ones. In order to test this hypothesis, the impact of different algorithms on the liquid composites' morphology, but also on the evolved devices' physical memory, result stability and capacity to learn from past training should be tested.

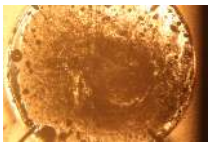
9.3.2 High Resolution Microscopy for Analysis

The microscopy set-up used in this work enabled the analysis of changes induced in the materials' morphology due to training. Images recorded on the material's surface with resolution were sufficient to link the rate of change in material morphology to the evolution the material's ability to process information. The question of whether an optimal solution have a specific structure remains open. In other words, is it possible to identify a path, either found, or evolved during training, which corresponds to the optimal solution to the computational problem at hand?

Using a thermal camera to record areas of heat within the material is a possible option. However a high resolution in necessary, as preliminary recordings have shown that it is difficult to distinguish a specific area of heat out of the overall change in the material's temperature. Other options are SEM of AFM imaging. However, these types of imaging require for the material to be in a solid state. It would not therefore be possible to compare the evolved SWCNT network with the pre-trained network. The area covered by the composite is also very large and it would only be possible to capture a fraction of the overall SWCNT structure.

A possible contender would be the high resolution ambient 3D microscopes with index-matched lenses used in biology. This would require minimal changes in the hardware set-up such as thinner microscope slides and composite films, but they would allow a high resolution 3D visualisation of the SWCNT network prior and post training. If a specific SWCNT-structure favouring optimal solutions was to be found using this microscopy technique, the morphology of the material could be recorded at each iteration (as is the case now) and fed back into the computer and used in the objective function to optimise training.



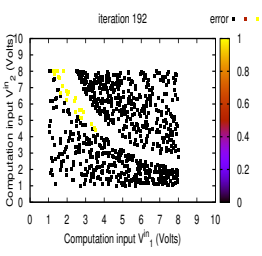


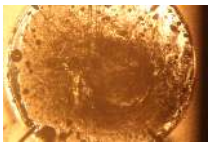
9.3.3 Analysis of SWCNT / Epoxy Composites Classifiers

The ability of evolved liquid SWCNT/epoxy classifiers to be solidified (cured) with negligible loss in classification accuracy was demonstrated in Chapter 6. Whilst this suggests that it will be possible for devices to benefit from the advantages of both the liquid and solid SWCNT-based composites, it also leads to further work. First of all, it would be interesting to verify whether the material in its liquid state presents all the characteristics of the SWCNT/LC composites: physical memory, retraining etc. In a second time, it would be interesting to test whether a schedule of training similar to that developed for algorithms run in conventional computers, could make possible the solving of very complex problems. Knowing the maximum number of solutions which the material can 'memorise'could also be the subject of investigations. Finally, different EiM frameworks could be tested against this material to determine if the accuracy issue is material-dependent or implementation-dependent.

9.3.4 Training Dynamic SWCNT-Based Composites with the RCiM Framework

Research presented in Chapter 5 suggested that training SWCNT/LC samples into simple binary data classifiers was a reproducible process. However, when tested against more complex problems such as the Iris and mammographic mass datasets, the solution quality deteriorated, with less reproducible results and classification accuracies lower than those obtain with state-of-the art machine learning implementations. Similarly, an issue with classification accuracy was observed in the evolved SWCNT/epoxy samples. In both cases, it would be interesting to investigate whether these materials could be trained using the RCiM framework. More importantly, the question is whether the use of this framework would improve classification accuracy, whilst producing devices with the characteristics observed in devices evolved with the classical EiM framework. A combination of high accuracy, low energy (no need for configuration voltages) and adaptive learning properties would make the evolved devices competitive with current technology. However, it is not certain that the structure-building process induced by RCiM training would result in those advantageous properties.





9.3.5 Replacing SWCNTs with Nanowires in Epoxy Matrix

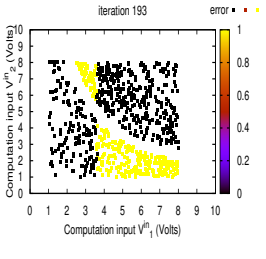
The main motivations behind the use of SWCNTs in the composites investigated were this material's interest in the field of electronics, combined with the recordable non-linear current / voltage behaviour it presented when dispersed in different polymers, LCs and epoxies. The impact on the computational capabilities of the SWCNT-baed composites of having both metallic and semi-conducting SWCNTs present in samples has not been investigated in EiM experiments. The semi-conducting nanotubes increase the resistivity of the composite, but it is unsure whether that has an impact on the solutions. Gold nanoparticle networks have been investigated in [3], demonstrating the capacity of composites with metallic only inclusions to be evolved via EiM. However, future work focusing on replacing the SWCNTs with other nanostructures such as Zinc Oxide or Gallium Arsenide nanowires would provide means of comparison between composites that include just metallic just semi-conducting or both metallic and semi-conductive wire-like structures. This has the potential of giving an indication regarding the contribution of the semi-conducting behaviour in the computational capabilities of composites for EiM. There is also the possibility that for the nanowire-based devices to be evolved to present more than one functionality.

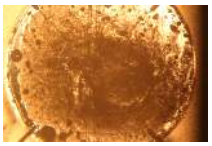
9.3.6 Exploring New Applications and Computational Problems

Classification and boolean logic problems have now been explored extensively within the field of EiM, irrespective of the framework, implementation or material used. Other problems have been suggested as more attractive for EiM investigations, such as natural language processing. However, other applications exist on the periphery of the typical computing problems used for machine learning *in silico*. The use of EiM to evolve devices able to identify volatile organic compounds could be envisaged. EiM training could also be used to produce devices capable of encrypting data. Preliminary investigations have demonstrated the potential for SWCNT-based devices to perform this latter task and the author believes it represents an extremely interesting avenue for research.

Bibliography

[1] S. L. Harding and J. F. Miller, "Evolution in materio: investigating the stability of robot controllers evolved in liquid crystal," *Evolvable Systems: From Biology to*

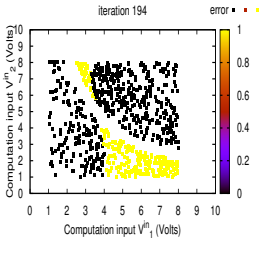




Hardware, pp. 155–164, 2005.

[2] J. Bird and E. Di Paolo, "Gordon pask and his maverick machines," in *The mechanical mind in history* (P. Husbands, O. Holland, and M. Wheeler, eds.), ch. 8, pp. 185–211, The MIT Press, 2008.

[3] S. Bose, C. P. Lawrence, Z. Liu, K. Makarenko, R. M. van Damme, H. J. Broersma, and W. G. van der Wiel, "Evolution of a designless nanoparticle network into reconfigurable boolean logic," *Nature nanotechnology*, vol. 10, no. 12, p. 1048, 2015.



Appendix A

Standard Substrate Cleaning Process

Standard microscope slides were cleaned in batches of 12 using the following standard washing procedure:

- Rinse with water (H20)
- 15 min in ultrasonic bath with water
- second rinse
- dry with nitrogen

The above steps repeated with:

- propan2ol
- acetone
- Decon 90
- DI water

Appendix B

Evolutionary Algorithm Performance on Benchmark Optimisation Function

The notation used for the algorithms' parameters, problem parameters and results in the tables B.1 and B.2 below are those presented in Chapter 3, Tables 3.1 and 3.2. Good values of algorithm and problem parameters, found using meta-optimisation, have been reported in [1, 2]. Some of these values are used here to evaluate the quality of the differential evolution (DE) algorithm and global particle swarm optimisation (PSO) developed for the purpose of the EiM. Implementations with lower number of iterations, dimensions and individuals were chosen since they better reflect the implementation that is used in the EiM experiments. The two algorithms were run on a 3.1GHz, 4 cores, 4GiB RAM desktop, this is different from the one used in [1, 2], but should not affect the experiments in other ways than time taken to compute, which is not compared here. Results obtained when solving of the Rosenbrock, Rastrigin and Ackley function are reported here. These three test functions are benchmarks for testing optimisation algorithms. They present a number of local minima, which makes them relatively difficult to solve, but each have one known global optimum. The Rosenbrock test function is given as

$$f_{Rb}(\mathbf{x}) = \sum_{d=1}^{D-1}(100 \cdot (x_{d+1} - x_d^2)^2 + (x_d - 1)^2), \tag{B.1}$$

with $d \in D$ the dimension index and the solution vector generally implemented as $\mathbf{x} \in [-100.0, 100.0]$. This function has a minimum at $f_{Rb}(\mathbf{x}^*) = 0$ for the optimum solution $\mathbf{x}^* = 1$. The number of dimensions to this problem is not set, with $\mathbf{x} = [x_1, ...x_D]^T$ and $-\infty \leq D \leq \infty$, instead, it can be considered a problem parameter. The Rastrigin test function is given as

$$f_{Rt}(\mathbf{x}) = \sum_{d=1}^{D}(x_d^2 + 10 - 10 \cdot cos(2\pi x_d)). \tag{B.2}$$

It has a minimum at $f_{Rt}(\mathbf{x}^*) = 0$ when the optimum solution $\mathbf{x}^* = 0$ is found. The number of dimensions to this problem is not set, with $\mathbf{x} = [x_1, ...x_D]^T$ and $-\infty \leq D \leq \infty$. This equation is generally implemented with $\mathbf{x} \in [-5.12, 5.12]$, which is also the case here. The Ackley test function is

$$f_{Ak}(\mathbf{x}) = 20 + e - 20 \cdot exp(-0.2\sqrt{\frac{1}{D}\sum_{d=1}^{D} x_d^2}) - exp(\frac{1}{D}\sum_{d=1}^{D} cos(2\pi x_d)), \tag{B.3}$$

with $\mathbf{x} \in [-30.0, 30.0]$. It has a minimum at $f(\mathbf{x}^*) = 0$ for the optimum solution $\mathbf{x}^* = 0$, with $\mathbf{x} = [x_1, ...x_D]^T$ for any value of D. Similar to the case of the Rosenbrock functino, D can be considered a problem parameter when solving the Rastrigin and Ackley functions.

B.1 Differential Evolution

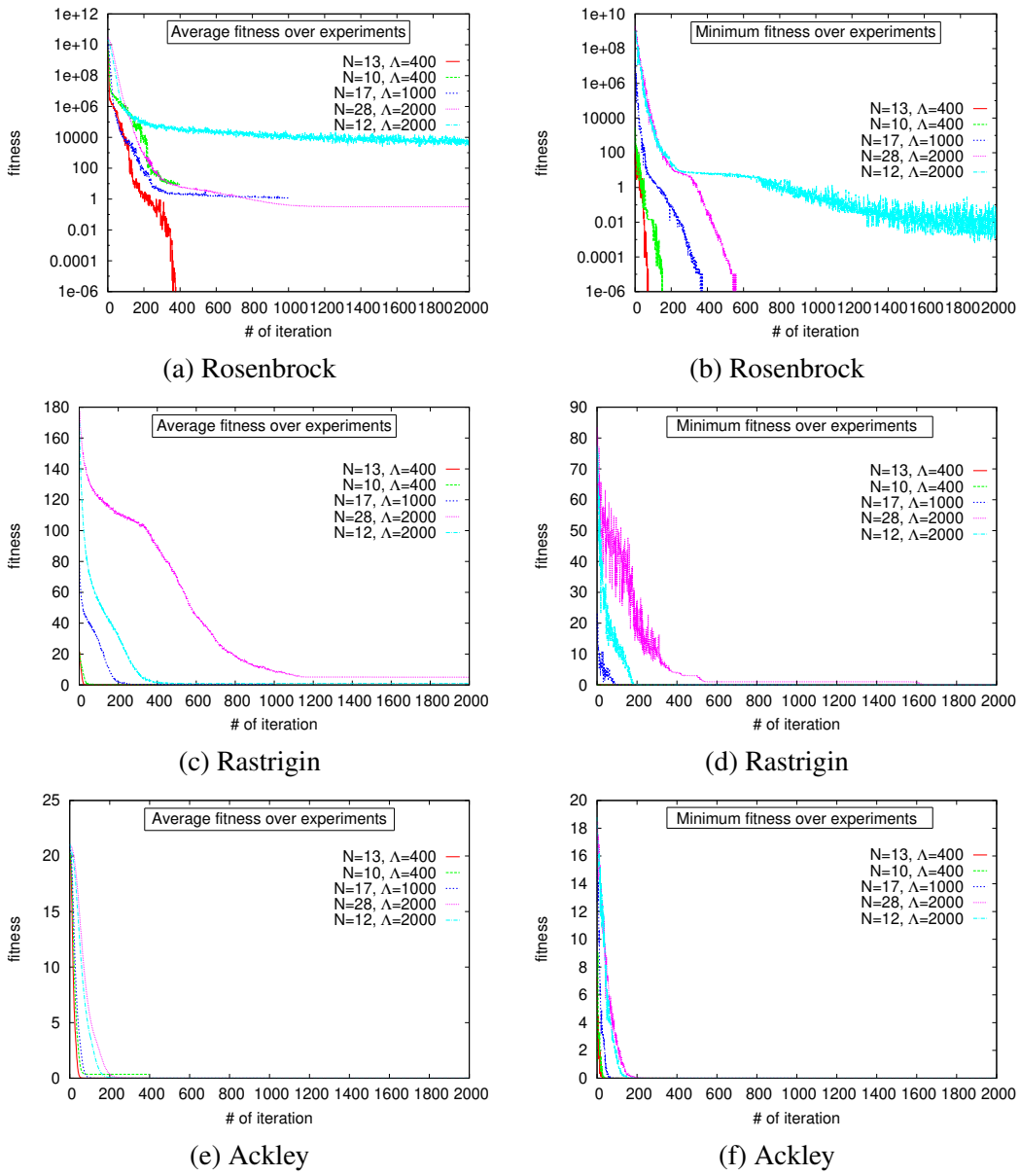
Table B.1 presents the DE and problem parameters reported as optimal for the Rosenbrock, Rastrigin and Ackley functinos in [1]. The third part of the table reports the best fitness value, $best^*$ and the iteration at which this best was obtained, λ^*, both averaged over fifty experiments. The last column reports the variance in best fitness, $\sigma(best)$, across the experiments, for each implementation and problem. Results reported in bold are optimal, i.e. the algorithm has converged towards the problem's global optimum in every experiment.

TABLE B.1: DE performance on three benchmark optimisation test functions, using the parameters and variables reported in [1].

	DE parameters			Problem parameters		Results		
	N	CR	F	D	Λ	$\overline{\lambda^*}$	$\overline{\text{best}}$	$\sigma(best)$
Rosenbrock	13	0.7450	0.9096	2	400	173.96	**0.0000**	**0.0000**
	10	0.4862	1.1922	2	400	219.24	8.6877	25.1843
	17	0.7122	0.6301	5	1000	886.06	1.1694	2.2591
	28	0.9426	0.6607	10	2000	1095.22	0.3189	1.0815
	12	0.2368	0.6702	10	2000	1975.98	6.2875	7.3890
Rastrigin	13	0.7450	0.9096	2	400	28.40	0.0199	0.1329
	10	0.4862	1.1922	2	400	41.34	0.0796	0.2699
	17	0.7122	0.6301	5	1000	205.84.04	0.3184	0.5781
	28	0.9426	0.6607	10	2000	930.90	4.9814	2.7109
	12	0.2368	0.6702	10	2000	425.90	0.8756	0.8117
Ackley	13	0.7450	0.9096	2	400	95.48	**0.0000**	**0.0000**
	10	0.4862	1.1922	2	400	119.02	0.3475	2.4284
	17	0.7122	0.6301	5	1000	189.40	**0.0000**	**0.0000**
	28	0.9426	0.6607	10	2000	500.90	**0.0000**	**0.0000**
	12	0.2368	0.6702	10	2000	407.10	**0.0000**	**0.0000**

The DE algorithm found the optimum solution to the Ackley test problem in all but one implementation. On the other hand, it was not able to always consistently find the optimum solution to the Rastrigin and Rosenbrock problems, irrespective of the implementations tested here. This consistent with the discussion reported in [1].

Across all problems, however, a global optimum was found in at least one experiment per implementation. This is better reflected in the convergence of the fitness function, illustrated in Figure B.1 for each problem and implementation. The fitness function values are averaged per iteration and over the fifty experiments, such each graph presents the convergence obtained across implementations for a given test function. Since the values of the fitness function start very high in the case of the Rosenbrock function, a log scale is used in Figures (a)-(d) for the sake of clarity. In the case of the other two functions, the y-axis follows a linear scale.



(a) Rosenbrock (b) Rosenbrock

(c) Rastrigin (d) Rastrigin

(e) Ackley (f) Ackley

FIGURE B.1: Convergence of the fitness function produced by a differential evolution algorithm over five different implementations averaged over fifty experiments for the (a) Rosenbrock, (c) Rastrigin and (e) Ackley benchmark test functions. Minimum fitness achieved by the PSO algorithm over five different implementations, per experiment, per iteration, for the (b) Rosenbrock, (c) Rastrigin and (b) Ackley benchmark test functions.

B.2 Particle Swarm Optimisation

Table B.1 presents the PSO and problem parameters reported as optimal for the Rosenbrock, Rastrigin and Ackley functions in [2]. The third part of the table reports the best fitness value, $best^*$ and the iteration at which this best was obtained, λ^*, both averaged over fifty experiments. The last column reports the variance in best fitness, $\sigma(best)$, across the experiments, for each implementation and problem. Results reported in bold are optimal, i.e. the algorithm has converged towards the problem's global optimum in every experiment.

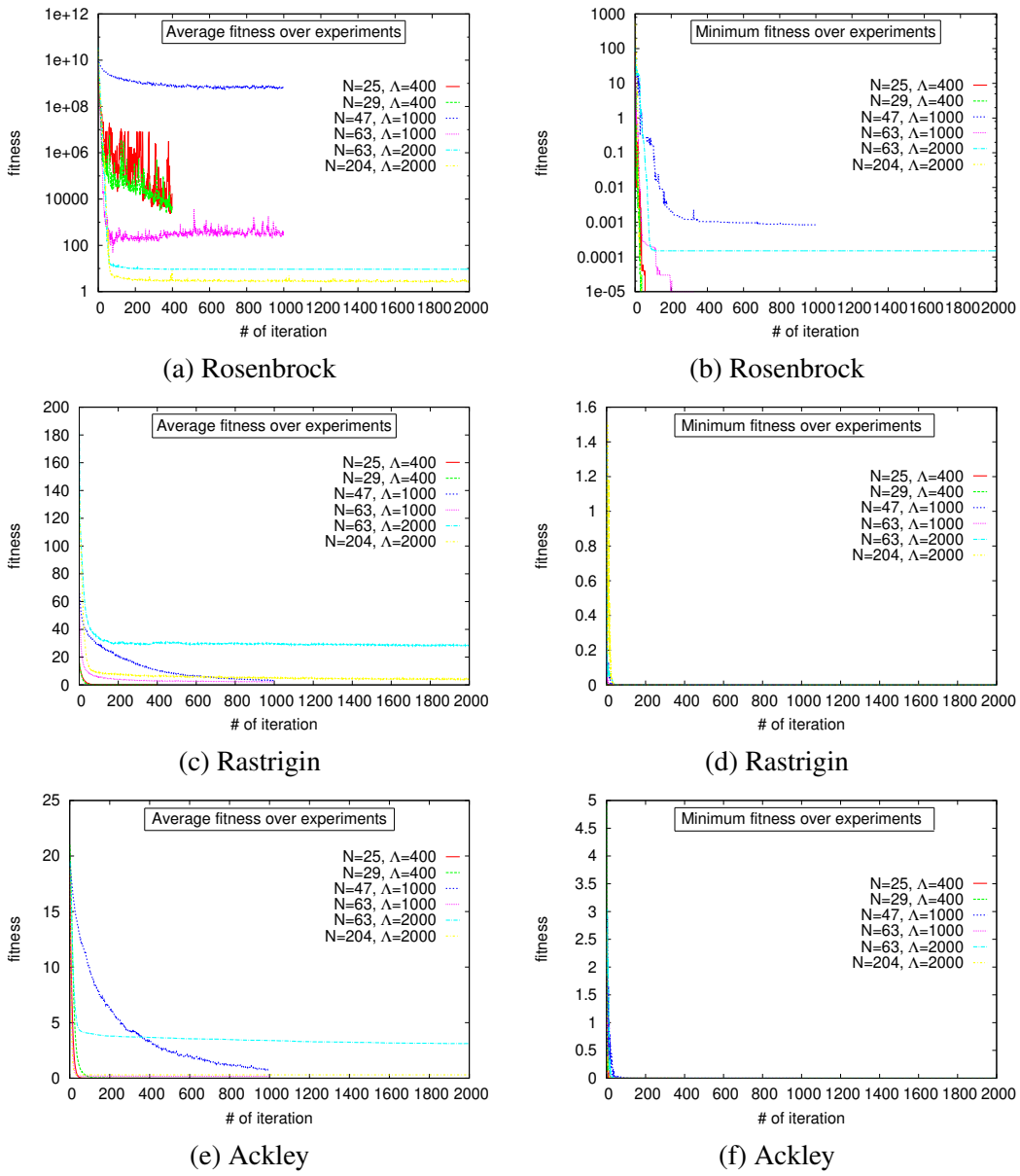
Across the three test functions, the global PSO used here was able to find the optimum solutions in every experiment for more than one implementation. The number of particles (individuals) did not appear to affect the quality of the solutions at low

TABLE B.2: Global PSO performance on three benchmark optimisation test functions, using the parameters and variables reported in [2].

	PSO parameters				Problem parameters		Results		
	N	ω	c_1	c_2	D	Λ	$\overline{\lambda^*}$	$\overline{\text{best}}$	$\sigma(best)$
Rosenbrock	25	0.3925	2.5586	1.3358	2	400	176.42	**0.0000**	**0.0000**
	29	−0.4349	−0.6504	2.2073	2	400	101.44	**0.0000**	**0.0000**
	47	−0.3593	−0.7238	2.0289	5	1000	986.04	3.37057	11.5747
	63	−0.1832	0.5287	3.1913	5	1000	865.3	0.51585	1.2828
	63	0.6571	1.6319	0.6239	10	2000	361.20	9.2954	16.4792
	204	−0.2134	−0.3344	2.3259	10	2000	1245.84	2.3825	2.3215
Rastrigin	25	0.3925	2.5586	1.3358	2	400	16.20	**0.0000**	**0.0000**
	29	0.3925	2.5586	1.3358	2	400	15.66	**0.0000**	**0.0000**
	47	0.3925	2.5586	1.3358	5	1000	204.68	**0.0000**	**0.0000**
	63	0.3925	2.5586	1.3358	5	1000	402.70	**0.0000**	**0.0000**
	63	0.6571	1.6319	0.6239	10	2000	727.50	0.4182	0.7981
	204	−0.2134	−0.3344	2.3259	10	2000	110.06	0.3383	1.6602
Ackley	25	0.3925	2.5586	1.3358	2	400	49.80	**0.0000**	**0.0000**
	29	0.3925	2.5586	1.3358	2	400	61.44	**0.0000**	**0.0000**
	47	0.3925	2.5586	1.3358	5	1000	464.6799	**0.0000**	**0.0000**
	63	0.3925	2.5586	1.3358	5	1000	65.560	0.1585	0.5459
	63	0.6571	1.6319	0.6239	10	2000	154.56	0.0000	0.0001
	204	−0.2134	−0.3344	2.3259	10	2000	57.92	0.2953	0.8106

dimensions (D=2), and it was always possible to find the problem's global optimum. However, when the number of dimensions increased (D≥5), i.e. with increasing problem complexity, PSO did not always converge towards the problems' optimum, and the convergence was dependent on the number of particles used. For higher dimensional implementations, a larger number of particles improved the algorithm's performance. This is consistent with the discussion reported in [2].

The convergence of the fitness function optimised by PSO, across each problem and implementation, is illustrated in Figure B.2 Similarly to DE, the fitness function values are averaged per iteration and over the fifty experiments, such each graph presents the convergence obtained across implementations for a given test function. Since the values of the fitness function start very high in the case of the Rosenbrock function, a log scale is used in Figures (a)-(d) for the sake of clarity. In the case of the other two functions, the y-axis follows a linear scale. As discussed in [2], and more generally in PSO-related literature [3, 4], the algorithm tends to converge quickly to an optimum, which can be either local or global. Once the algorithm has converged, it is difficult for it to escape this optimum, which, if it is local, means that the global optimum of the problem at hand will never be reached.



(a) Rosenbrock (b) Rosenbrock

(c) Rastrigin (d) Rastrigin

(e) Ackley (f) Ackley

FIGURE B.2: Convergence of the fitness function produced by a global particle swarm optimisation algorithm over five different implementations averaged over fifty experiments for the (a) Rosenbrock, (c) Rastrigin and (e) Ackley benchmark test functions. Minimum fitness achieved by the PSO algorithm over five different implementations, per experiment, per iteration, for the (b) Rosenbrock, (c) Rastrigin and (b) Ackley benchmark test functions.

Bibliography

[1] M. E. H. Pedersen, "Good parameters for differential evolution," tech. rep., Technical report, Hvass Computer Science Laboratories, 2010.

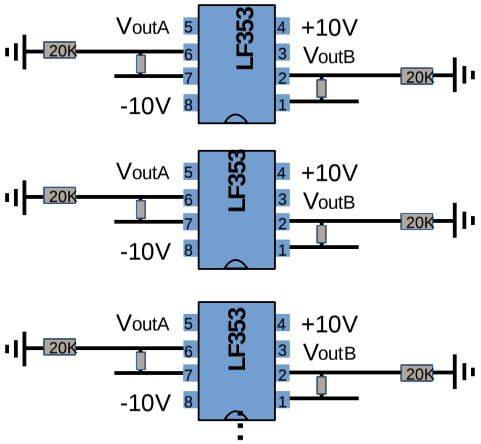
[2] M. E. H. Pedersen, "Good parameters for particle swarm optimisation," tech. rep., Technical report, Hvass Computer Science Laboratories, 2010.

[3] Y. S. R. C. Eberhart, "Comparing nonlinear inertia weights and constriction factors in particle swarm optimization," *2000 IEEE Conference*, 2000.

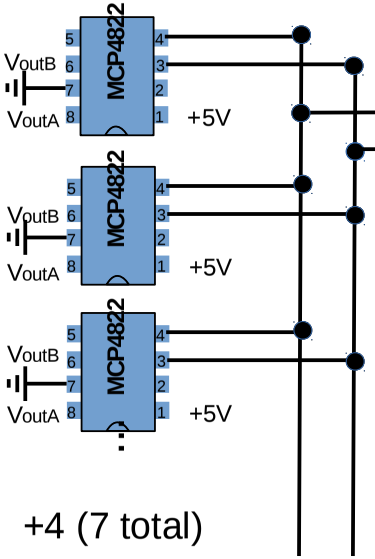
[4] A. P. Engelbrecht, *Computational intelligence: an introduction*. John Wiley & Sons, 2007.

Appendix C

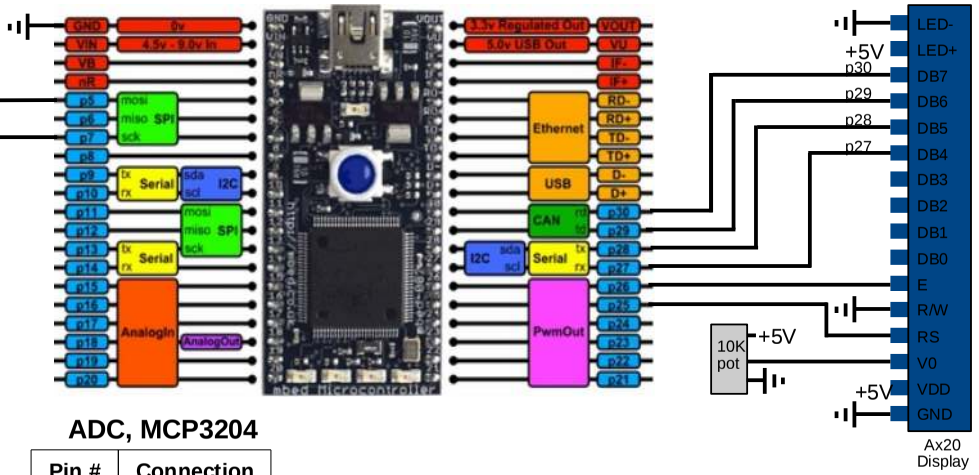
Evolvable Motherboard: Details



+4 (7 total)



+4 (7 total)



OPAMP, LF353

Pin #	Connection
1	1OUT (b# sample)
2	1IN- (20K GND)
3	1IN+ (VoutB)
4	Vcc-
5	2IN+ (VoutA)
6	2IN- (20K GND)
7	2OUT (b# sample)
8	Vcc+

DAC, MCP4822

Pin #	Connection
1	VDD (mbed +5V)
2	CS (mbed p8,9,...)
3	SCK (p7)
4	SDI (p5)
5	LDAC (! connected)
6	VoutB
7	Avss (GND)
8	VoutA

ADC, MCP3204

Pin #	Connection
1	CH0
2	CH1
3	CH2
4	CH3
5	
6	
7	GND
8	CS (p20)
9	Din (p5)
10	Dout (p6)
11	CLK (p7
12	GND
13	14
14	+5V (V0)